Computer Science and Information Systems

Published by ComSIS Consortium

Volume 10, Number 1
January 2013

Computer Science and Information Systems

Vol 10, No 1, January 2013

ComSIS is an international journal published by the ComSIS Consortium

ComSIS Editorial Office:
University of Novi Sad, Faculty of Sciences,
Department of Mathematics and Informatics
Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia
Phone: +381 21 458 888; **Fax:** +381 21 6350 458
www.comsis.org; Email: comsis@uns.ac.rs

Volume 10, Number 1, 2013
Novi Sad

Computer Science and Information Systems

ISSN: 1820-0214

ComSIS Journal is sponsored by:

Ministry of Education, Science and Technological Development of Republic of Serbia - http://www.mpn.gov.rs/

Computer Science and Information Systems

AIMS AND SCOPE

Computer Science and Information Systems (ComSIS) is an international refereed journal, published in Serbia. The objective of ComSIS is to communicate important research and development results in the areas of computer science, software engineering, and information systems.

We publish original papers of lasting value covering both theoretical foundations of computer science and commercial, industrial, or educational aspects that provide new insights into design and implementation of software and information systems. ComSIS also welcomes survey papers that contribute to the understanding of emerging and important fields of computer science. In addition to wide-scope regular issues, ComSIS also includes special issues covering specific topics in all areas of computer science and information systems.

ComSIS publishes invited and regular papers in English. Papers that pass a strict reviewing procedure are accepted for publishing. ComSIS is published semiannually.

Indexing Information

ComSIS is covered or selected for coverage in the following:

- Science Citation Index (also known as SciSearch) and Journal Citation Reports / Science Edition by Thomson Reuters, with 2011 two-year impact factor 0.625,
- Computer Science Bibliography, University of Trier (DBLP),
- EMBASE (Elsevier),
- Scopus (Elsevier),
- Summon (Serials Solutions),
- EBSCO bibliographic databases,
- IET bibliographic database Inspec,
- FIZ Karlsruhe bibliographic database io-port,
- Index of Information Systems Journals (Deakin University, Australia),
- Directory of Open Access Journals (DOAJ),
- Google Scholar,
- Journal Bibliometric Report of the Center for Evaluation in Education and Science (CEON/CEES) in cooperation with the National Library of Serbia, for the Serbian Ministry of Education and Science,
- Serbian Citation Index (SCIndeks),
- doiSerbia.

Information for Contributors

The Editors will be pleased to receive contributions from all parts of the world. An electronic version (MS Word or LaTeX), or three hard-copies of the manuscript written in English, intended for publication and prepared as described in "Manuscript Requirements" (which may be downloaded from http://www.comsis.org), along with a cover letter containing the corresponding author's details should be sent to official Journal e-mail.

Criteria for Acceptance

Criteria for acceptance will be appropriateness to the field of Journal, as described in the Aims and Scope, taking into account the merit of the content and presentation. The number of pages of submitted articles is limited to 25 (using the appropriate Word or LaTeX template).

Manuscripts will be refereed in the manner customary with scientific journals before being accepted for publication.

Copyright and Use Agreement

Computer Science and Information Systems

Volume 10, Number 1, January 2013

CONTENTS

Papers

Special Section: Engineering of Computer Based Systems

Special Section: Information Technologies in Medicine and Rehabilitation

EDITORIAL

This issue of Computer Science and Information Systems consists of 13 regular articles and two special sections: “Engineering of Computer Based Systems,” guest-edited by Bernard Schätz, which contains six articles that represent expanded versions of papers selected from the 19th Annual IEEE International Conference and Workshops on the Engineering of Computer Based Systems (ECBS), and “Information Technologies in Medicine and Rehabilitation,” guest-edited by Goran Devedžić, which brings forth four papers that describe new developments in the interdisciplinary field of biomedical engineering. We would like to use this opportunity to thank the guest editors, as well as article authors and reviewers, for helping to bring this diverse issue of ComSIS to our readers.

In the first regular article, “WebMonitoring Software System: Finite State Machines for Monitoring the Web,” Vesna Pajić et al. present a system based on finite-state machines that successfully solves two problems regarding information search on the Web: enabling effective complex search queries that transcend keywords, and accessing Web-page content that would otherwise be hidden due to crawling limitations and time lags.

Dušan Okanović et al., in “SLA-Driven Adaptive Monitoring of Distributed Applications for Performance Problem Localization,” describe DProf – an adaptive approach to application-level monitoring of software systems which allows changing the instrumentation of software operations in monitored distributed applications at runtime. This is achieved by specifying performance objectives in service level agreements (SLAs) and using call tree information to detect and localize problems in application performance.

“A Scalable Multiagent Platform for Large Systems” by Juan M. Alberola et al. introduces a new multi-agent platform developed at the level of the operating system, facilitating high efficiency and scalability to large populations of agents that require fast messaging services, agent group management, and security checks.

The article “Validation of Schema Mappings with Nested Queries” by Guillem Rull et al. tackles the problem of validating XML schema mappings, focusing on nested relational schemas. Validation is performed through reasoning on schemas and mapping definition, by encoding the given mapping scenario into flat database schema, and reformulating property checks as query satisfiability problems.

In “Accessibility Algorithm Based on Site Availability to Enhance Replica Selection in a Data Grid Environment,” Ayman Jaradat et al. present a replica selection algorithm for data grid environments that considers site availability in addition to data transfer time, providing better estimates of response time compared to existing approaches which do not take site availability into account.

Raka Jovanovic and Milan Tuba, in their article “Ant Colony Optimization Algorithm with Pheromone Correction Strategy for the Minimum Connected Dominating Set Problem,” give special attention to the initial condition of the colony optimization (ACO) algorithm for the minimum connected dominating set problem (MCDSP), also adding a pheromone correction strategy. The two innovations avoid entrapment in local optima, as well as reduce complexity of the ACO algorithm.

“Ontological Model of Legal Norms for Creating and Using Legislation,” by Stevan Gostojić, Branko Milosavljević and Zora Konjović, presents a formal model of legal norms modeled in OWL. Unlike existing approaches that model legal norms by formal logic, rules or ontologies, the approach presented in this article is intended for semiautomatic drafting and semantic retrieval and browsing of legislation.

George Lagogiannis, Nikos Lorentzos, and Alexander B. Sideridis, in “Indexing Moving Objects: A Real Time Approach,” tackle the problem of reducing the I/O bottleneck when indexing moving objects by minimizing the number of I/Os in such a way that queries concerning the present and past positions of the objects can be answered efficiently. The authors propose two approaches that achieve an asymptotically optimal number of such I/Os, based on the assumption that the primary memory suffices for storing the current positions of the objects.

In “Multi-sensor Data Fusion Based on Consistency Test and Sliding Window Variance Weighted Algorithm in Sensor Networks,” Jian Shu et al. tackle the problem of reduced accuracy of sensor data due to zero offset and decreased stability of wireless sensor networks by proposing an algorithm for detecting abnormal data based on consistency tests and sliding window variance. The article shows that amending or removing abnormal sensor data using the proposed method results in better precision compared to existing approaches.

Zhang Yong-Xin, Li Qing-Zhong and Peng Zhao-Hui, in “A Novel Method for Data Conflict Resolution using Multiple Rules,” examine the issue of conflict resolution during data integration by considering the interplay of data conflict resolution on different attributes, instead of focusing on resolving conflicts between single attributes. They propose a two-stage procedure based on Markov Logic Networks, and show that the proposed approach can significantly improve the accuracy of data conflict resolution in real-world scenarios.

The following three articles are significantly extended and improved versions of papers presented at the 5th International Conference on Information Technology (ICIT 2011) that underwent the regular submission and reviewing procedure.

The article “Ontology-Based Architecture with Recommendation Strategy in Java Tutoring System” by Boban Vesin et al. presents Protus 2.0, the new version of the tutoring system for learning basic concepts of Java programming language, focusing on its modular architecture where each Protus 2.0 component is represented by a specific ontology, and course personalization achieved through learner style identification and content recommendation.

“A Viewpoint of Tanzania E-Commerce and Implementation Barriers” by George S. Oreku, Fredrick J. Mtenzi and Al-Dahoud Ali discusses the prospects of e-commerce implementation, participation, motivation and opportunity in developing countries like Tanzania, with large domestic markets and potentials for the development of the agricultural sector. The paper concludes that Tanzanians have the ability to participate in e-commerce, but with the need for improving the national image by introducing trust and discipline.

Finally, Slavica Aleksić et al., in the article “A Design Specification and a Server Implementation of the Inverse Referential Integrity Constraints,” present an approach to the automated implementation of inverse referential integrity constraints (IRICs) within the SQL Generator tool developed as a part of the IIS*Studio development environment. The paper describes the algorithms for insertion, modification and deletion control, and illustrates them through an example of generated procedures/triggers.

Editor-in-Chief
Mirjana Ivanović

Managing Editor
Miloš Radovanović

EDITORIAL

Special Section: Engineering of Computer Based Systems

The following six articles represent expanded versions of selected high quality papers presented at the 19th Annual IEEE International Conference and Workshops on the Engineering of Computer Based Systems (ECBS), April 11-13, 2012, Novi Sad, Serbia.

Tomas Potuzak in his paper entitled “Methods for Division of Road Traffic Network for Distributed Simulation Performed on Heterogeneous Clusters” presents two road network division methods for heterogeneous clusters, MBFSMTL (Modified Breadth-First Search Marking of Traffic Lanes) and GAMTL (Genetic Algorithm Marking of Traffic Lanes), that are based on their counterparts originally designed for homogenous clusters. Described methods consider the different computational powers of nodes in the heterogeneous cluster and divide the computational load among the road traffic sub-networks according to a benchmark test that directly utilize the road traffic simulation in order to obtain the most relevant information about the speeds of nodes in the cluster.

The article “Modeling and Visualization of Classification-Based Control Schemes for Upper Limb Prostheses” by Andreas Attenberger and Klaus Buchenreider proposes a model of the classification process for upper-limb prostheses including a subsequent simulation, validation and visualization of the prosthesis control scheme. Their experiments show that classification schemes based on electromyographic data can be improved significantly by integrating additional data from NIR (near-infrared) sensors. In addition to the classification process, the behavior of prosthesis is demonstrated through the simulation of a 3D hand model that is controlled by the classifier output.

The next paper “On Task Tree Executor Architectures Based on Intel Parallel Building Blocks”, by Miroslav Popović, Miodrag Đukić, Vladimir Marinković and Nikola Vranić, deals with the problem of applying parallel programming techniques based on Intel Parallel Building Blocks to a class of service components within SOA based industrial systems. The paper presents two novel Task Tree Executor (TTE) architectures, the first one that is based on Intel Threading Building Blocks (TBB) library, and the second one based on Intel Cilk Plus library. The novel architectures execute TTE tasks as TBB tasks and Cilk strands, respectively, rather than the local operating system threads, providing better multicore CPU utilization.

Xi Wu, Huibiao Zhu, Yongxin Zhao, Zheng Wang and Si Liu, in the paper entitled “Modeling and Verifying the Ariadne Protocol Using Process Algebra”, apply the process algebra method known as Communicating Sequential Processes to model and analyze route discovery in the Ariadne protocol. The formal model of the Ariadne protocol is implemented in the model checking tool PAT in order to verify security properties of the protocol. The verification results show that there is a defect in the protocol, which may lead to fake routing attacks.

In the article “System Design for Passive Human Detection using Principal Components of the Signal Strength Space”, Bojan Mrazovac, Milan Z. Bjelica, Dragan Kukolj, Branislav M. Todorović and Saša Vukosavljev propose a device free detection method of human presence based on principal component analysis (PCA) of the radio signal strength variations. The method exploits the fact that the presence of a human within a wireless network range results in significant signal strength variations at the receiver. Experimental results of presented research show that PCA inputs, given in a form of raw RSSI (Received Signal Strength Indicator) samples, provide more accurate detection of human presence, than the inputs which describe the dispersion of the signal.

Finally, “Support for End-to-End Response-Time and Delay Analysis in the Industrial Tool Suite: Issues, Experiences and a Case Study” by Saad Mubeen, Jukka Mäki-Turja and Mikael Sjödin presents the implementation of two state-of-art real time analysis techniques in the form of individual plug-ins for the existing industrial tool suite Rubus-ICE. The paper discusses the experience gained while transferring theoretical research results to the industrial tool suite. In a case study, implemented plug-ins are used to analyze the model of the autonomous cruise control system.

Guest Editor of Special Section
Bernard Schätz
Technical University Munich, Germany

EDITORIAL

Special Section: Information Technologies in Medicine and Rehabilitation

Information and emerging medical technologies brought immense progress in the broad fields of bioengineering, clinical engineering, and medical and health informatics, particularly during the last two decades. The evident interdisciplinary nature of these fields is making clear that modern medicine cannot exist without modern diagnostic and therapeutic equipment that are the result of synergy of medical and engineering knowledge. Improvements in human health and clinical practice are direct consequences of coupling between novel biomedical methods and applications, and advances in information technologies. For example, 3D modeling and analysis of musculoskeletal and vascular systems, medical implants, clinical equipment, therapeutic and rehabilitation devices, tissue modeling, etc. On the other side, medical informatics comprises of clinical knowledge, information processing and communication through development of new algorithms, knowledge representation, and data analysis. In addition, health informatics essentially contributes to the storage, retrieval, and optimal use of the biomedical information, data, and knowledge. For example, electronic medical records organize patients' health and clinical information and data, enabling the improvement of health care quality, efficiency and data collection.

Having that ubiquitous information technologies integrate the engineering sciences with the biomedical sciences and clinical practice is the key motivation for organizing the Special Section of the ComSIS journal devoted to the Information Technologies in Medicine and Rehabilitation. New developments and advances in the interdisciplinary scientific field of bioengineering presented in this issue of the ComSIS journal span over different sub disciplines, which include hearing instruments, embedded biomedical devices, gait analysis and arterial stiffness analysis.

Tessendorf et al. present a newly developed wireless multimodal hearing system, which is a context-aware device that analyzes the acoustic environment in order to automatically adapt sound processing to the user's current hearing wish. In order to satisfy user's different hearing wishes in the same acoustic environment, the authors investigated additional modalities to sound that can provide the missing information, which determines the user's hearing wish, to improve the adaption. Their platform takes into account additional sensor modalities such as the user's body movement and location.

Motivated by telemedicine and home care systems Stojanović et al. present a methodology and techniques that implement discrete wavelet transform in

low-complexity fixed point embedded architectures of biomedical devices. These are intended to be a low-cost, miniature and telemetry capable to overcome the distance barrier between the doctor and patient, e.g. remote vital sign monitors. They implemented their methodology to a “systems on chip” device, consisting of a single microprocessor/microcontroller. The approach resulted in an increased processing speed, minimized memory requirement and decreased power consumption.

Filipović et al. in their study offer an innovative and robust approach to assess 3D kinetics of a knee and the stress and strain distributions in the knee-based subject-specific biomechanical models of the human knee joint. For the study they used the MRI imaging and the measured kinematic data. The paper presents an algorithm for contour recognition and 3D reconstruction of the bones, cartilages and meniscuses geometry obtained by the MRI scans. These 3D models, together with the measurement data are the inputs for the computational analysis, using the finite element method, that determine the stress and strain distribution at different body postures during the gait analysis. Such an approach opens new avenues for an objective assessment of pre- and post-operation knee functioning.

Peulić et al. use a finite elements method to model effects of the arterial stiffness using the different signal patterns of the pulse transit time (PTT). They measured the PTT signal of several different breathing patterns of the three subjects and applied finite element model of the straight elastic artery to compute arterial elastic behavior, as well as the simplex optimization method for fitting procedure to estimate Young’s module of the arterial stiffness. The result suggests that approximately the same value of Young’s module can be fitted for specific subject with different breathing patterns, which validate this methodology for possible noninvasive determination of the arterial stiffness. The proposed method allows the implementation of screening diagnostics. For clinical usage it is sufficient to register equal duration of an ECG signal and the distal arterial pulse, which are carried out with the non-invasive methods by means of widely available monitoring devices.

Organizing the Special Section devoted to the Information Technologies in Medicine and Rehabilitation would not be possible without genuine encouragement and support of Prof. Mirjana Ivanovic, Editor-in-Chief of ComSIS, and Prof. Ivan Lukovic, Journal’s Vice Editor-in-Chief, who have kindly accepted the request to publish the new achievements in applications of information technologies in the field of biomedical engineering.

Special Section Guest Editor
Goran Devedžić

DOI:10.2298/CSIS110918036P

WebMonitoring Software System: Finite State Machines for Monitoring the Web

Vesna Pajić[1], Duško Vitas[2], Gordana Pavlović Lažetić[2], and Miloš Pajić[1]

[1] University of Belgrade, Faculty of Agriculture, Nemanjina 6,
11080 Zemun, Belgrade, Republic of Serbia
svesna@agrif.bg.ac.rs, paja@agrif.bg.ac.rs
[2] University of Belgrade, Faculty of Mathematics, Studentski trg 17,
11000 Belgrade, Republic of Serbia
vitas@matf.bg.ac.rs, gordana@matf.bg.ac.rs

Abstract. This paper presents a software system called WebMonitoring. The system is designed for solving certain problems in the process of information search on the web. The first problem is improving entering of queries at search engines and enabling more complex searches than keyword-based ones. The second problem is providing access to web page content that is inaccessible by common search engines due to search engine's crawling limitations or time difference between the moment a web page is set up on the Internet and the moment the crawler finds it. The architecture of the WebMonitoring system relies upon finite state machines and the concept of monitoring the web. We present the system's architecture and usage. Some modules were originally developed for the purpose of the WebMonitoring system, and some rely on UNITEX, linguistically oriented software system. We hereby evaluate the WebMonitoring system and give directions for further development.

Keywords: finite state automata, finite state transducers, software, web monitoring, electronic dictionaries, web search

1. Introduction

The problem of effective search for some particular piece of information is quite current, because of the tremendous amount of information available on the World Wide Web. Natural Language Processing (NLP) and Computational Linguistics (CL) play a dominant role in attempts to solve this problem. Depending on the structure of an electronic text, properties of the language it is written in, and user requirements, NLP and CL have different levels of success. Although very fast and powerful search engines already exist, there are still problems that remain unsolved. In our research, we focused on the two of them.

First, there is a problem of "invisible web" [1], i.e., not having access to some content on the web through search engines. Almost every modern

search system (Google[1], Yahoo[2] etc.) consists of several sub-systems, the most important being the sub-systems for crawling, indexing, and ranking web pages. Since search systems are faced with a huge number of web pages to download and process, each subsystem imposes certain limitations. For the search engine's crawler to find a web page, it is necessary either that the web developer has notified the system of its existence on the web by registering the page URL to the search engine, or that the crawler has visited some other web page containing the hyperlink that points to it. Otherwise, the page remains "invisible" to the search system, and therefore to users. In addition, given that resources of every search system are limited, each crawler has certain limitations that keep crawling process within the limits of available resources. Some search engines limit the total number of pages in the index and drop the old pages when there are new ones, while others limit the frequency of repeated visits to pages. Whenever the search engine decides to limit the search process, the part of the information remains unavailable to users.

There is also a time difference between the moment some content is uploaded to the web and the moment the crawler finds it. This is a major problem for pages that frequently change their content, such as daily news or different forums. Some search engines, including Google, allow the author of a website to reduce this interval, i.e. to increase the frequency at which the crawler visits the site in order to better respond to customer requirements. Another way of overcoming the problem of dynamic content is the concept of RSS ("Really Simple Syndication"), which implies that the author of a particular web site edits and maintains a list of changes on the web site. This list is called "RSS feed". Customers interested in following the changes on the web site can access this list automatically by using a special program known as RSS aggregator. In either case, the visibility of the content depends on the website author. If the author has not provided an RSS feed, nor reduced the interval of the crawler's visits to the page, the users can do no more but personally and regularly check the contents of a particular website in searching for specific information.

The second problem is the way search engines queries are composed, which is often not adequate. Regardless of the facts that Internet users come from different countries, have different areas of interest and levels of education, speak different languages, and have different needs for information, they are all facing the same or similar forms when querying search engines. Mostly, it is a HTML form for keyword search, sometimes with advanced options that allow users to further limit their search. There is no way for a user to formulate some more complex queries.

This problem is even more evident in case of morphologically rich languages, such as Serbian language. For example, if users want to find web pages about Serbian national football team, they would be interested in documents that contain some of the phrases:

1 http://www.google.com

2 http://www.yahoo.com

reprezentacija Srbije („national team of Serbia");
naša reprezentacija („our national team");
reprezentativci Srbije („football players of national team of Serbia");
fudbaleri Srbije („football players of Serbia");
tim „orlova" („eagles" team);
naš nacionalni tim ("our national team");
tim Radomira Antića ("Radomir Antic's team");
srpski fudbaleri ("Serbian football players");
srpska ekipa ("Serbian team");
srpska reprezentacija ("Serbian national team").

Even more, the documents containing any of the inflectional forms of the above phrases are also relevant for the user. General-purpose search engines do not allow making such queries. In recent years, Google has made significant efforts to improve its search process and to bring it to customers, so it returns results where keywords appear in inflected forms as well. This is a good attempt to improve search, but the big problem is that users have no control over this process.

In our research, we focused on solving the above-mentioned problems: improving the way of formulating queries, which will allow for describing more complex context of information, and enabling the access to the content on the Web in the shortest possible time interval. The proposed solution uses finite state automata as search queries, which allows users to describe very complex contexts and phrases they wish to find on web pages. We overcame the second problem by designing a special system that supports the concept of monitoring the web, over which the user has full control. As a result, we have developed the software system called WebMonitoring, which works as a client application on a user's computer. It has its own crawling sub-system that allows a user to set a seed URL and a depth level of crawling, and therefore to monitor one page, one part of a web site or the whole web site. Users describe information they want to find by graphs representing finite state automata or transducers. Those graphs are used as a query for the search. Users can set the way they wish to be alarmed if some information occurs on the monitored web page, as well as the interval of repeated checks.

2. State of the Art in Monitoring the Web

In attempts to improve the process of searching for information on the WWW, different tools for searching and monitoring the web have been developed. Depending on their architecture and functionality, as well as the problems they focus on, they can be divided into two groups.

The tools in the first group focus on monitoring the web. They are all designed primarily for notification if a web page has been changed; there is no possibility to search for some complex queries, except queries based on keywords using Boolean operators. A user can set the frequency of downloads, but it is often limited to some minimum interval (in most cases it is

12 hours interval). Changes occurring on the page in less than 12 hours may not be noticed by the system. Some of the tools use regular expressions, but only to restrict the monitored web page content. Examples of such tools are online systems ChangeDetect[3] and WebSite Watcher[4].

The ChangeDetect system is an online tool for monitoring web pages. For each page a user wants to monitor it is possible to set several parameters: frequency of downloads, regular expression filtering, events causing alerting the user and so on. Still, the minimum monitoring interval is 12 hours. Changes occurring on the page in less than 12 hours may not be noticed by the system. Moreover, the system is designed primarily for notification if a web page has been changed. There is no possibility to search for complex queries, nor the possibility to monitor the entire site. While a user can assign multiple pages to be monitored in the control panel, the monitoring system monitors only the pages with listed URL within each process.

WebSite-Watcher is the software designed to track changes on any number of web pages. With this tool it is possible to monitor all formats of electronic text, including even password-protected pages. Moreover, this tool allows a user to monitor changes of binary files in the sense of changing the size or the date the file was changed. However, even this tool does not support setting up complex queries. WebSite-Watcher uses regular expressions, but only to restrict the monitored web page content. A user is enabled only to search for the occurrence of a keyword or a phrase.

The second group of available tools contains tools that focus on making queries for the search. They are often linguistically oriented, and one of the best known is WebCorp system [2]. WebCorp is a software tool designed by the Department of Research and Development of English Language at the University of Birmingham. It is intended primarily for linguistic research, but it can be used for search as well. A user is allowed to search for a specific word, a phrase, or a pattern. Patterns can be formed by combining the operator *, which replaces any sequence of characters in an expression, square brackets and the OR operator. Although this tool represents a significant improvement since it allows describing complex phrases or patterns, it still does not solve the problem since it relies upon results from the existing search engines. Therefore, there is still much information on the web that remains inaccessible by this tool.

A free online concordance service, GlossaNet [3], is a software tool that solves many of the problems related to information search on the web. It is intended for search into dynamic Web corpora. Users define a corpus by selecting RSS feeds in a pre-selected pool of sources. The GlossaNet crawler regularly visits these sources in order to generate a dynamic corpus. A user can register one or more search queries on his/her dynamic corpus, which are represented with finite state automata and graphs. Those queries will be reapplied to the corpus every time it is updated and new concordances will be recorded for the user. The GlossaNet greatly improves the process of search,

[3] http://www.changedetect.com
[4] http://www.aignes.com

but its main disadvantage is the way it makes corpora. The GlossaNet relies upon RSS feed, and therefore upon a web site's owner and his/her decision about what is to be updated. If the owner of a web site does not provide information about some change in page content, the user will not be able to access that information.

3. WebMonitoring Software System

1.1. Theoretical Background

Finite State Machines in NLP. Finite state machines (automata and transducers) are used in many fields of computational linguistics. Their use is justified from the standpoint of linguistics, as well as from the standpoint of computer science. From the linguistics point of view, finite state machines are adequate for describing relevant local phenomena in language research and for modeling some parts of natural language, such as its phonology, morphology, or syntax. Some examples of adequate representation of different linguistics phenomena by finite state machines are given in [4]. From computer science point of view, the use of finite state machines is motivated by time and space efficiency. Time efficiency is achieved by using deterministic finite state machines. The output of the deterministic machines depends mostly on the size of the input, so they are considered to be optimal ([5] and [6]). Space efficiency is achieved by minimizing deterministic machines [7].

Finite state machines can be very complex and difficult to maintain, which leads to some problems in practice. For example, if someone tries to describe the language syntax by finite state machine, the corresponding graph would be very immense, and finding some particular information, such as noun phrases, would be time consuming and impractical. So, instead of one big graph, we use a collection of sub graphs. This method has a strong theoretical background in the theory of Recursive Transition Networks (RTN). RTN are extension of context free grammars ([8] and [9]). The arcs in RTN are labeled with corresponding grammars, while the states are labeled arbitrarily. There are several computer tools for linguistic research based on FSM and RTN ([10], [11] and [12]). Detailed review of theoretical and practical use of finite state transducers in natural language processing is given in [13], [14], [15], [16], [17] and [18].

The Concept of Monitoring the Web. The concept of web monitoring has emerged from the need for automating certain actions taken by user in order to be notified of changes that occur on a particular web page or site. In practice, there are often situations when a user visits a web site expecting that some event occurred on it, without being interested in the rest of the content of the web site. Examples of such events are announcement of the results of

some competition, electronic message with the specific content or from a specific person, the appearance of news about a particular topic, and so on. In such cases, the user regularly, with the schedule that he or she believes is optimal or possible at a time, visits the web site of interest looking for the event. The software for web monitoring automates this searching process and simulates the actions that the human would take.

1.2. Architecture of the WebMonitoring System

The WebMonitoring software system is developed in the order to overcome problems in search process defined in the Section 1. It is written in the Java programming language and consists of several sub-systems:

- the system for making queries – it is based on the Unitex [11] software system
- the management system
- the crawler
- the system for text post-processing
- the alarming system
- the graphical user interface

The description of the software architecture is given in Figure 1.

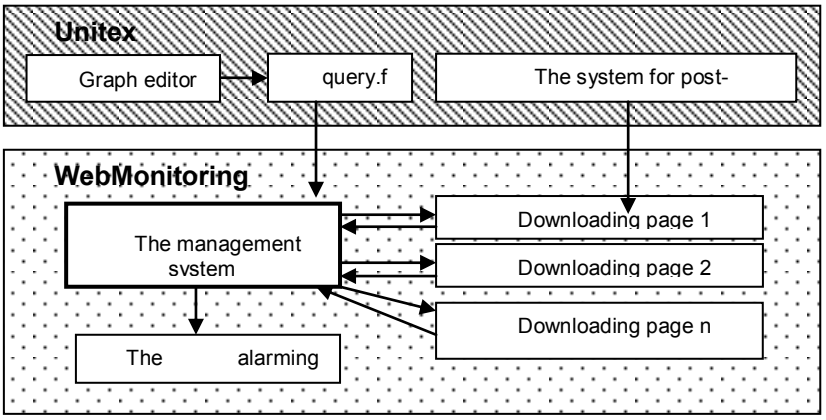


Fig 1. Architecture of the WebMonitoring system

The management system, the crawler, the alarming system and the graphical user interface were developed and written by the authors for the purpose of the WebMonitoring system, while we used the Unitex software and some of its components for querying and post-processing the text.

A user creates a graph that describes an event of interest using the Unitex system. This graph is passed as an input to the management system. Using the graphical user interface of the WebMonitoring software system, the user

sets the URL of the page or website which is being monitored, adjusts parameters, such as dynamics of the visits, the depth of the crawl process (relative to the page from which the crawl will start), the way of alarming, and so on. After that, the management system runs one separate programming thread for each monitoring process set in the management system. The web pages found in the crawling process are saved locally. The system for post-processing analyzes and processes these pages, and tries to find patterns corresponding to the graph. When a pattern is found (and event occurred), the system notifies the user.

The System for Making Queries. The WebMonitoring system uses graphs produced by the Unitex software system [11] as search queries. Unitex is a collection of programs developed for analyzing text written in natural languages, and for applying different linguistic resources and tools to the text. It is an open source software with a very good, functional, and user-friendly graphical interface. Apart from its well-designed graphical user interface for creating graphs, one of the main advantages of the Unitex software is the possibility to use linguistic resources, such as electronic dictionaries and grammars.

Electronic dictionaries contain simple and compound words, together with their lemmas and the set of grammatical codes. They are constructed by teams of linguists for different languages (for English language [19] and [20], for French language [21] and [22], for Serbian language [23] and [24]). Unitex uses electronic dictionaries in DELA format, where each entry is a line of text terminated by a new line, which conforms to the following syntax:

```
apples,apple.N+conc:p
```

The first word (`apples`) is an inflected form of the entry and it is mandatory. In the former example it is followed by the canonical form (lemma) of the entry. This information may be left out if the canonical form is the same as the inflected form. The following sequence of codes (`N+conc`) gives the grammatical and semantic information about the entry. In the former example, code `N` stands for noun, and `conc` indicates that this noun designates a concrete object. The label `p` stands for “plural”.

Although Unitex can process text in different languages, in the first version of the WebMonitoring software it is assumed that Serbian language will be used. For that reason electronic dictionaries for Serbian language are used ([23] and [24]) in the WebMonitoring modules for post-processing the text.

After applying dictionaries and grammars to the text, Unitex creates separate files with simple words, compound words, and unrecognized words. Those files are used in the search process, so one can refer to the dictionary entry from the Unitex by using lexical masks. For example, a user can use the query `<be.V>` that matches all entries having `be` as canonical form and the grammatical code `V`. Thus all occurrences of the verb *to be* (*am*, *is*, *being* etc.) will be recognized by this query. Beside lexical masks, a user can use

morphological filters. For example, the filter <<ism$>> matches all words that end with "*ism*" (conservatism, racism, etc.).

By applying lexical resources to the text, as well as combining lexical masks and morphological filters, users can make graphs that correspond to very complex queries. Those graphs in Unitex may have two formats, the format `.grf`, which is intended for the design phase of graphs, and the format `.fst2`, which is compiled version of graphs, intended for further processing and applying to a text.

The graph that corresponds to the phrases about the Serbian national team described in the Section 1 is shown in the Figure 2.

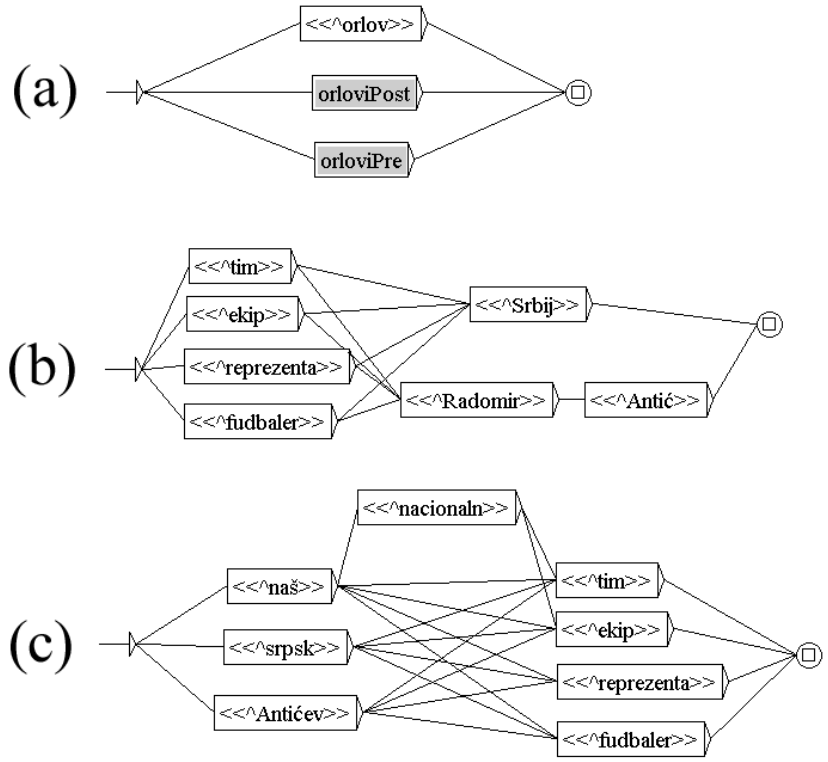


Fig. 2. (a) RTN for describing the phrases corresponding to the Serbian national team; it contains calls to sub-graphs *orloviPost* and *orloviPre*; (b) *orloviPost* is a sub-graph for describing the terms in which the affiliation is given after the noun; (c) *orloviPre* is a sub-graph for describing the terms in which the affiliation is given before the noun

The content of the corresponding `.fst2` file, which is used as a search query, is as follows:

```
0000000003
-1 orlovi
: -2 1 -3 1 1 1
t
f
-2 orloviPre
: 9 2 8 1 6 1
: 5 3 4 3 3 3 2 3
: 7 4 5 3 4 3 3 3 2 3
t
: 5 3 4 3
f
-3 orloviPost
: 5 1 4 1 3 1 2 1
: 11 3 10 2
t
: 12 2
f
%<E>
%<<^orlov>>
%<<^reprezenta>>
%<<^fudbaler>>
%<<^tim>>
%<<^ekip>>
%<<^srpsk>>
%<<^nacionaln>>
%<<^Antićev>>
%<<^naš>>
%<<^Srbij>>
%<<^Radomir>>
%<<^Antić>>
f
```

The `.fst2` format is strictly defined by the Unitex software. The first line represents the number of graphs that are encoded in the file. Lines containing the number and the name of the graph identify the beginning of each sub-graph. In the above file, those are the lines `-1 orlovi`, `-2 orloviPre` and `-3 orloviPost`. The following lines describe the states of the graph. If the state is final, the line starts with `t` character, and with `:` character if not.

For each state, the list of transitions is a sequence of pairs of integers. The first integer indicates the number of the label or sub-graph that corresponds to the transition. Labels are numbered starting from 0. Sub-graphs are represented by negative integers. The second integer represents the number of the result state after the transition. In each graph the states are numbered starting with 0. By convention, state 0 is the initial state.

From the standpoint of the WebMonitoring users, the Unitex system and its interface for creating graphs represent a system for making queries, i.e. for describing an event a user wants to be notified of. Using the Unitex system, a

user creates a graph that describes the event of interest, and then compiles it into the `.fst2` format. This file contains all the necessary information about the event of interest and, as such, it represents the input data for the WebMonitoring system.

The Management System. The management system of the WebMonitoring software consists of several Java classes, and it represents the central point of the overall system. It is designed so as to enable a user to run more than one independent monitoring process.

Every monitoring process is defined by the following attributes:

- *URL* – a web page URL from which the crawl and the search start;
- *graph* – a location of .fst2 file which describes the searched phrases;
- *levels* – an integer that defines the depth of crawl; this attribute is explained in more details later;
- *alarm* – a string attribute that defines the way a user should be alarmed if the event occurred. There are two possibilities: sending an e-mail message and saving the page on the local hard disk. This attribute has the following form: `address; location`, where `address` is an e-mail address for sending the message, and `location` is a directory path for saving the page. If any of these parts of the alarm attribute equals *null*, there will be no alarming of that kind;
- *timeInterval* – an integer that represents a time interval (in milliseconds) between two repeated search processes.

There is an instance of the class *MonitoringProcess* for every monitoring process in the programming code. The class *MonitoringProcess* extends the *Thread* class in Java, i.e. it is a runnable thread. In that way the independent and parallel monitoring of different pages and searching for different queries are allowed.

When the application is started, the main window of the management system will open (Figure 3). It will show the monitoring processes that were started during the previous run in the table, if any. A user can select some process from the table to view or change its characteristics. From this window the user can also delete, make a new, start or stop a process.

The Crawler. After the user has started a monitoring process, the crawler starts a crawl from the URL that is set in the process properties. The crawling process depends on the value of the *levels* parameter. If the value of the parameter is 1, the system should take only this page.

Otherwise, when one page is downloaded from the Internet, it is analyzed to find other hyperlinks on it. The system stores found hyperlinks in the crawl queue and sends them back to the crawler, with the crawling level decreased by one. The crawl process is stopped when all the pages with the level greater than 0 have been processed.

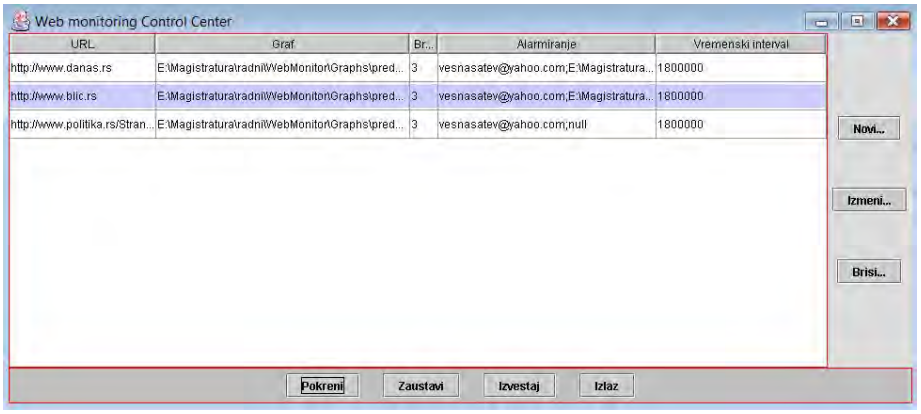


Fig. 3. The control center of the management system

The crawler of the WebMonitoring system is designed to satisfy the following basic principles of a good crawler [25]:

- efficiency - a number of hyperlinks a crawler needs to process increases exponentially with the number of visited pages increasing, so the system must be able to handle the list of addresses in efficient manner from the point of memory usage;
- duplicates – the crawler needs to add to the address list only those addresses that have not been visited, i.e. to recognize the addresses of pages that have already been processed and not to add them to the list;
- politeness – the crawler must comply with the directives contained in the *robots.txt* file on the web server, and to avoid downloading too many pages from a website, so its functionality should not be threatened;
- avoiding the traps - crawler must be able to recognize and avoid sites that are created with the intent to cause an infinite loop for crawlers visiting them.

The crawling sub-system consists of several classes, all belonging to the package *crawler.driller*:

- class *_GO_PARAMS* keeps all the important parameters for the crawling process, such as the seed URL, the number of levels for crawling, the array of the web addresses from the starting page to the current one, etc.;
- class *DrillingQueue* defines a dynamic data structure for keeping the queue of web URL's which have to be downloaded and processed;
- class *PathFinder* keeps information about the path crawler used to get some page;
- class *Driller* starts downloading pages from the seed URL, regarding the number of levels (the *levels* parameter).

The starting class of the crawling sub system is the *CrawlerShell* class, which performs all the necessary adjustments of parameters and runs the crawler. There are many parameters within this class intended for configuring the system for downloading pages. For the purpose of the WebMonitoring

system, some of these parameters have default values that cannot be changed by the user. Thus, for example, the value of the parameter *timeout* is 60 seconds, which means that the crawler sub system will wait for up to one minute for downloading a page from the Internet. Similarly, the maximum time to download pages from one site (in a monitoring process) is defined by the *maxTimeout* parameter, and it is set to 20 minutes (1200 seconds). The declaration section of the CrawlerShell class is shown in the following code:

```
  /** Class for downloading pages starting from the
seed URL */
  public class CrawlerShell{

  // default argument values
    static String hunt = null;
    public static boolean thisSiteOnly = false, silent
= false;
    public static int maxTimeout = 1200, timeout = 60;
    public static int maxlinks = 1000, maxretries = 3;
    public static int levels = 0, sleepParamMS = 500;
    static boolean analyze = true, passedAskMe = false;
    static boolean flash = false, displayLinks = true;
    static magicPath = false;
    static long timeStart = System.currentTimeMillis();
    static boolean logHttp = false, hideUrls = true;
    ......
```

After adjusting the parameters, the crawler for the given URL is started. The text found on every web page in this process is sent to the post-processing system for further analysis, i.e. for the graph search.

The System for Text Post-processing and Alarming. When a web page is downloaded from the Internet, first it is necessary to prepare the text it contains, and then to search for the appropriate event defined by the search graph. This task is performed by the system for post-processing and its class *MonitoredText*. Since the Unitex's external program *Locate* has the central place in the search process, the text from the page is prepared in accordance with the requirements of this program.

The text found on the web page is saved in a text file and stored in a temporary directory. This file is the starting file in the text processing. Although text on web pages is coded differently, most websites in Serbian language nowadays use UTF-8 encoding. UTF-8 (*8 bit Unicode Transformation Format*) encodes each Unicode character as a variable number of 1 to 4 octets, where the number of octets depends on the integer value assigned to the Unicode character. Since each character in the range of U+0000 through U+007F is represented as a single octet, UTF-8 is a very efficient encoding schema of text documents in which most characters are US-ASCII. This is also the reason why this encoding became dominant for electronic mail and web documents, and therefore WebMonitoring system

assumes that the page is encoded in UTF-8. The text found on the page is saved in a text file using methods from the class *fr.umlv.unitex.io.UnicodeIO*. This class is specially designed for the Unitex system purposes, so every file made during the post-processing can be read and processed adequately.

The first step in the post-processing is normalization of the text. Normalization is a process in which the normalization of separators of the text is made, although it is possible to normalize the text on the basis of some other specific rule. Separators in a text are space characters, tabulators, and new lines. When the text is taken from a web page, it is possible that it contains several separators placed side by side in a sequence. These kinds of sequences of separators are being replaced by one space character in the process of normalization. The process of normalization is performed by the Unitex's external program *Normalize*.

The second step of post-processing is the process of text tokenization, i.e. breaking the text up into lexical units. In order to tokenize the text, the Unitex's external program *Tokenize* is called within the class *MonitoredText*.

The electronic dictionaries are applied to a text by the Unitex's external program *Dico*. This step provides the possibility to use morphological and lexical filters in search queries.

After the text is processed in the above described manner, the next step is to search for the patterns described by the user's graph. The search is done by the Unitex's external program *Locate*, which applies a particular automaton or transducer described by a graph to the text and creates an index of found phrases. The program Locate creates two files: `concord.ind` with the references to occurrences found in the text, and `concord.n`, with a number of occurrences and a percentage of recognized tokens. These two files are saved in the working directory, and are used by the sub system for alarming. Method *foundConcordances()* of the class *MonitoredText* uses the file `concord.n` to read the number of occurrences of the searched phrases. If this number is greater than 0, i.e. if the phrases that correspond to the search graph are found in the text, the method *alert()* of the class *MonitoredText* is called, and the user is informed about the event. The way of alarming depends on the values of attributes `email` and `location`. The user can choose to be alerted by an e-mail message, or just to locally save the web page [26].

4. A Use Case

The WebMonitoring software system can be used for different tasks, such as press clipping, detecting spam messages by monitoring electronic mailboxes, management of various documents collections, and so on. We will describe one possible use of the WebMonitoring system.

A user wishes to find all articles related to the current president of the Republic of Serbia, Mr. Boris Tadic, which are or will be published in daily newspapers. The user takes the following steps.

Step 1. Defining the Event to be Searched for. The event to be searched for is an occurrence of phrases regarding Mr. Boris Tadic. Some of the phrases (in Serbian language) are: "*predsednik Tadić*" ("*the president Tadic*"), "*predsednik Srbije*" ("*the president of Serbia*"), "*predsednik B. Tadić*" ("*the president B. Tadic*"), "*predsednik Boris Tadić*" ("*the president Boris Tadic*"), "*Boris Tadić*", as well as their inflected forms ("*predsednika Tadića*", "*Borisu Tadiću*" etc.).

Step 2. Describing the Event by a Graph. The user uses Unitex and creates a graph that describes the defined event. This task can be performed in many different ways, and it depends on user's skills and available resources. One possible way of creating the graph is by using morphological filters. An example is given in the Figure 4. It is necessary to save and compile graph in Unitex, so the file with the extension `.fst2` is created.

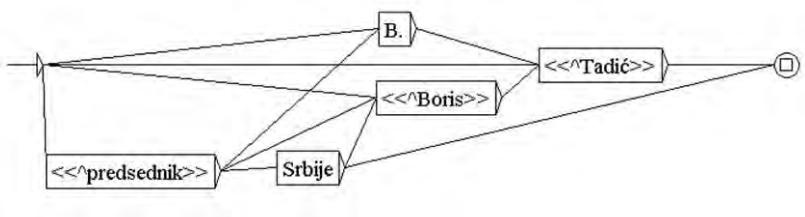


Fig. 4. The query automaton

Step 3. Choosing the Content to Monitor. The user chooses web pages or web sites he/she wishes to monitor. Having in mind that the user wishes to find news articles, he/she chooses official web sites of several daily news papers in Serbia (http://www.danas.rs, http://www.blic.rs and http://www.politika.rs/Stranice/33.lt.html) as starting points of the monitoring process.

Step 4. Creating Monitoring Processes in the WebMonitoring System. In the WebMonitoring system, the user creates a process for each web site he/she wishes to monitor. For each process the user sets URL, number of levels for the crawl, location of the graph describing the event, the way of alarming, and the interval for repeating the process. The main window of the application will appear as shown in the Figure 5.

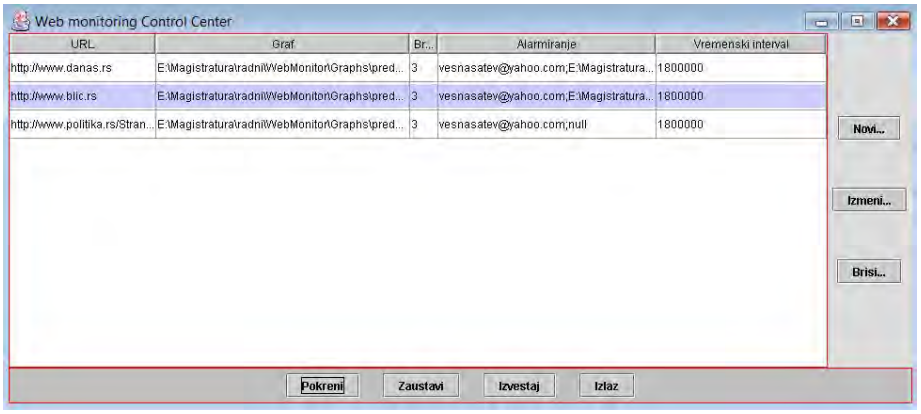


Fig. 5. Control center with three monitoring processes

Step 5. Starting the Processes. The user selects the process in the table showing processes and starts it by choosing the appropriate button. The crawling and monitoring process starts and works in the background. The user can see the progress by choosing the button "*Izvestaj*" ("*Report*"). If the phrase that matches the graph is found on some page, the user is alerted either by e-mail or the page is saved locally on the user's computer.

5. Results and Evaluation of the WebMonitoring System

The case described in Section 4 was used to evaluate the WebMonitoring software system. Since we wanted to evaluate both the possibility to process complex queries and the possibility to access web content in the short time after it appears on the web, we conducted the similar searches with two existing services: GlossaNet, as one of the most powerful, linguistically oriented search system, and Google. We had to change some of the search parameters slightly depending on the requirements and the architecture of these two services, but we tried to keep the search process as uniform as possible.

We monitored two web sites, the official website of the Serbian newspapers *Blic* (http://www.blic.rs) and a popular forum *Krstarica* (Cruiser) in Serbian language (http://forum.krstarica.com). Both web sites have a very dynamic content that frequently changes. Nevertheless, the *Blic* web site provides a RSS feed, while *Krstarica* forum does not.

The results of the evaluation test significantly differ for the two web sites. Comparative characteristics of the three systems are presented in Table 1. The summary of monitoring the *Blic* web site is given in the Table 2, and the summary of monitoring the *Krstarica* forum is given in the Table 3. Since Google retrieves documents (web pages) instead of occurrences of the searched phrases, the number of pages on which the searched phrases are

found is given in the Table 2 and 3. The results are further discussed in the following text, and the results from WebMonitoring system are given in Appendix A.

Table 1. Comparison between GlossaNet, Google and WebMonitoring

System	GlossaNet	Google	WebMonitoring
Type of monitoring	Automatic	Manual	Automatic
Ability to set the monitoring interval	No	No	Yes
Type of query	Finite state graph	Keywords	Finite state graph

Table 2. The results of monitoring the *Blic* web site

System	GlossaNet	Google	WebMonitoring
Number of concordances found by a system	14	850*	51
Most recent result found	Unknown	6 hours	3 minutes

* The number of pages on which the concordances are found; the actual number of concordances is even greater

Table3. The results of monitoring the *Krstarica Forum* web site

System	GlossaNet	Google	WebMonitoring
Number of concordances found by a system	-	43*	14
Most recent result found	-	4 hours	15 minutes

* The number of pages on which the concordances are found; the actual number of concordances is even greater

Search queries. The WebMonitoring software system and the GlossaNet service can process Unitex graphs as search queries, so we used the graph given in the Figure 4. On the other hand, Google can process only keyword searches, so we query Google with: `tadić OR "predsednik tadić" OR "predsednik srbije" OR "boris tadic"`. In that way we were able to control inflectional forms of words with the WebMonitoring and the GlossNet systems, while we did not have any control, nor the possibility to look for different forms of the same word with Google unless we explicitly put it in the search query. In other words, the WebMonitoring and the GlossaNet enable users to search with complex queries, while Google does not.

Monitoring a web site. After defining the search queries, it was necessary to set up web sites to monitor. The content of the chosen web sites was used as a text corpus to search. The GlossaNet system uses a RSS feed from a web site as a mechanism for creating and refreshing a text corpus used in a search process. A user can choose from a predefined list of websites, or create its own corpus, but only for a web site with a RSS feed. After that, the GlossaNet system monitors the chosen site and alerts the user after the monitoring process is over, sending a message with occurrences found. We chose Blic corpus to be monitored during 24 hours. Since Krstarica forum has not got a RSS feed, we could not monitor it with the GlossaNet.

The Google system has its own crawling process in which it downloads and indexes the downloaded pages. Many parameters of the Google's crawling process are automated and recalculated during the process, so a user has no control over the process. Therefore, it is necessary for the user to manually query the Google search system in intervals he/she thinks to be optimal. In our evaluation test, we did two hours check during 24 hours, passing the described search query restricted on the two chosen web sites (adding text "`site:blic.rs`" and "`site:forum.krstarica.com`" to the search query). We also added a 24 hours restriction in order to narrow the search and to achieve a better comparison of results.

The WebMonitoring software system provides the greatest possibilities in terms of setting and controlling the monitoring process. We choose to monitor web sites http://www.blic.rs and http://forum.krstarica.com, with the crawling depth set to 3 during the same 24 hours.

The time interval between two visits to the web site content. The GlossaNet system uses a RSS feed from a web site as a mechanism for refreshing a text corpus used in a search process. The links from RSS feed are visited on a regular basis, but the time period between two visits to links from a RSS feed is defined by the GlossaNet system for corpora building and cannot be changed by a user.

Each time the Google crawling system crawls a URL, it detects whether the resource has changed since the previous crawl. If the resource changed, the change interval is shortened. If the resource did not change, the change interval is lengthened. In that way, a user cannot affect the time interval between two visits. In our evaluation test, we monitored web sites with very dynamic content, and since the Google crawls them with high frequency, the recently added pages from these two web sites were available to Google search. The problem arises when monitoring web pages that do not change their content in a longer time period, and then suddenly change.

The WebMonitoring software system allows a user to set up the time interval, depending on his/her expectations. In our evaluation test, we used 30 minutes as a time interval between two crawls.

The content of a web site "visible" to search systems. Given the way the GlossaNet and the Google download pages from a web site, some contents

remain invisible to them, and therefore to a user. For example, the GlossaNet service downloaded (and searched) only links provided in the RSS feed. Thus, dozens of pages from the Blic web site were not processed. Moreover, Krstarica web site could not be processed at all since it does not provide RSS feed. Therefore, the GlossaNet system was useless in monitoring process of Krstarica forum web site.

In comparison to Google, the WebMonitoring system processed much less pages, but it was expected regarding the architecture of the systems. The Google's crawler and indexing system is far ahead of other crawlers, and we had no intention to compete with Google in it. The advantage of WebMonitoring over Google is in accessing web content in a short time after it appears on the web. In our evaluation test, Google reported one web page as the most recent result, giving the time "26 minutes ago", which would mean that the page was found by Google at 14:50 (Figure 6). On the other hand, the time of publishing given on the page was 8:08 AM 29.02.2012. with changes at 8:55 AM (Figure 7). This means that there is a 6 hours time gap from the time this web page was published on the web to the time Google found it. In our evaluation test, the WebMonitoring system found this page 3 minutes after it was published. In the worst case this time gap can be 30 minutes since the monitoring process is being repeated every 30 minutes. Moreover, a user can additionaly reduce the interval, if needed.

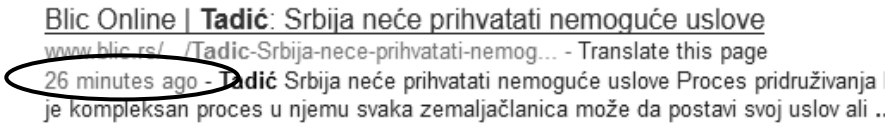


Fig. 6. The most recent Google result: 26 minutes ago (29.2.2012. 14:50)

Fig. 7. The headline as it appeared on the Blic web site, published at 08:08 AM 29.02.2012.

Some additional remarks about the WebMonitoring software system are:

- the WebMonitoring crawling system works with high success; there were no errors or situations that some web page could not be downloaded;
- the recognition of phrases corresponding to the graph is complete, i.e. all phrases that existed in the page content and that correspond to the graph have been recognized (we used a sample of 30 pages from *Blic* web site for this testing);
- the speed performance of the system is satisfying having in mind its purpose, although some improvements can be made. This relates primarily to the use of RAM memory instead of saving the changes to the hard disk. Unitex and its external programs record every change of a text on the hard disk, and this practice was continued in the WebMonitoring system, but it significantly slows down the system. Also, the system can be speeded up by using more programming threads. In the current version, one monitoring process is executed within one programming thread, while the inside operations of downloading and processing web pages run sequentially;
- during monitoring of the web site http://www.blic.rs, the event occurred in more than 45% of all web pages. The reason for that is not a significant number of articles about the president of Serbia, but the design of the site. On every web page of the web site there is a section with current news, showing the same news. In the future version of the system this problem should be solved, i.e. the system must be able to recognize the same context on the different pages.

6. Conclusions

This paper considers the improvement of information search process in terms of making more complex queries and access to content of web pages in a short period after their posting on the web. As a solution for complex querying, we suggest using finite state machines. We used finite state machines through the software system Unitex for making queries, but also for the post-processing of the text. We designed and developed the software system called WebMonitoring, which has integrated a subsystem for crawling web pages. With such a system users can do their own crawl and search web pages they wish, independently from the common search engines.

Furthermore, the system WebMonitoring has features that allow user to create, maintain and control processes of monitoring web pages or sites. The system simulates and automates actions a human would take in the process of looking up for some event (a phrase occurrences) on a page.

Since search queries are passed to the system as Unitex graphs, representing finite state automata, this system is not intended for use of a casual user. A basic understanding of finite state automata is required so a user could successfully describe a complex context of searched phrases. Nevertheless, the Unitex' graphical user interface for creating and modifying graphs is user friendly and very intuitive, so any user could easily be trained

to use it. Additionaly, the system could be expanded with modules for automatic transforming regular expressions into Unitex graphs. In that way a user will be able to choose between regular expressions and graphs, depending on his/her skills.

The first version of the WebMonitoring software system should be considered as a demonstration how it is possible to integrate lexical word processing programs with a concept such as web monitoring. Although the current version of the WebMonitoring software system is fully functional and shows positive effects of applying finite state machines to the search process, it is necessary to make some improvements in future versions of the system.

These enhancements are primarily related to elimination of possible errors in the code and upgrading performance and speed of the program. In addition, a user should have more control over the process itself in terms of selecting the language or deciding whether or not to apply dictionaries to the text. The WebMonitoring software system is a general-purpose system. In future versions it is possible to modify the system so as to be specialized for specific types of text (such as medical, technical, etc.), or for special purposes, such as monitoring electronic mailboxes, or search for a specific product in a database accessible from the Internet. We expect that these specialized versions of the WebMonitoring software system will be more efficient.

Nevertheless, the WebMonitoring software system is important because it demonstrates a way of overcoming some problems in the process of information search. It also gives directions for use of linguistic tools in the search process and transcends the limitations in accessing information of the existing search engines.

Acknowledgments. The work presented has been financially supported by the Ministry of Science and Technological Development, Republic of Serbia, Project No. 178006.

References

1. Sherman, C., Price, G.: The Invisible Web: Uncovering Information Sources Search Engine Can't See, Information Today Inc. (2005)
2. A. Kehoe, A. Renouf: WebCorp: Applying the Web to Linguistics and Linguistics to the Web, WWW2002 Conference, Honolulu, Hawaii (2002).
3. C. Fairon, GlossaNet: Parsing a web site as a corpus, Lingvisticae Investigationes, John Benjamins Publishing Company, Volume 22, Number 2, pp. 327-340(14) (2000)
4. M. Gross, D. Perrin, Electronic Dictionaries and Automata in Computational Linguistics, in Proceedings of LITP Spring School on Theoretical Computer Science Saint-Pierre d'Oleron, France, May 25.-29. (1987)
5. D. Vitas, Prevodioci i interpretatori: Uvod u teoriju i metode kompilacije programskih jezika, Matematički fakultet, Belgrade, Republic of Serbia (2006)
6. D. Jurafsky, J. H. Martin, Speech and language processing, Prentice-Hall Inc., 2000.

7. A. V. Aho, J. E. Hopcroft, J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison Wesley, Reading, MA (1974)
8. J. M. Sastre, M. Forcada, Efficient parsing using recursive transition networks with output, In Zygmunt Vetulani, editors, 3rd Language & Technology Conference (LTC'07). 5-7 October 2007. pp. 280–284 (2007)
9. J. M. Sastre, Efficient Parsing Using Filtered-Popping Recursive Transition Networks, Lecture Notes in Computer Science. vol. 5642. pp. 241–244 (2009)
10. B. Olivier, M. Constant, E. Laporte, Outilex, plate-forme logicielle de traitement de textes ecrits. In Proceedings of TALN'06. Leuven, Belgium, UCL Presses universitaires de Louvain (2006)
11. S. Paumier, Unitex 1.2 User Manual, Université de Marne-la-Vallée. http://www-igm.univ-mlv.fr/˜unitex/UnitexManual.pdf (2006)
12. M. D. Silberztein, Dictionnaires électroniques et analyse automatique de textes : le système INTEX. Paris: Masson. (1993)
13. F. Casacuberta, E. Vidal, D. Picó, Inference of finite-state transducers from regular languages, Pattern Recognition, Volume 38, Issue 9, pp.1431-1443 (2005)
14. N. Friburger, D. Maurel, Finite-state transducer cascades to extract named entities in texts, Theoretical Computer Science 313, pp 93 – 104 (2004)
15. J. R. Hobbs, D. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, M. Tyson, FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text, In Roche E. and Y. Schabes, eds., Finite-State Language Processing, The MIT Press, Cambridge, MA, pages 383-406 (1997)
16. A. Kornai, Extended finite state models of language, Cambridge University Press (1999)
17. E. Roche, Finite state transducers: parsing free and frozen sentences, Extended finite state models of language, Cambridge University Press, pp. 108.-120 (1999)
18. E. Roche, Y. Schabes, Finite-state language Processing, The MIT Press, (1997)
19. A. Chrobot, B. Courtois, M. Hammani-Mc Carthy, M. Gross, K. Zellagui. Dictionnaire electronique DELAC anglais : noms composés. Technical Report 59, LADL, Université Paris 7, (1999)
20. A. Savary. Recensement et description des mots composés - méthodes et applications, Thèse de doctorat. Université de Marne-la-Vallée, (2000)
21. B. Courtois and Max Silberztein, editors. Les dictionnaires électroniques du français.Larousse, Langue française, vol. 87, (1990)
22. J. Labelle, Le traitement automatique des variantes linguistiques en français: l'exemple des concrets, Lingvisticæ Investigationes, 19(1), Amsterdam - Philadelphia: John Benjamins Publishing Company, pp.137–152 (1995)
23. C. Krstev, D. Vitas, Corpus and Lexicon - Mutual Incompletness, in Proceedings of the Corpus Linguistics Conference, Birmingham, (2005)
24. D. Vitas, C. Krstev, I. Obradović, Lj. Popović, G. Pavlović-Lažetić, Processing Serbian Written Texts: An Overview of Resources and Basic Tools, in Workshop on Balkan Language Resources and Tools, 21 Novembar 2003, Thessaloniki, Greece, pp. 97-104 (2003)
25. Baroni, M., Bernardini, S.: BootCaT: Bootstrapping corpora and terms from the Web. In: Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-2004), Lisbon (2004)
26. V. Pajic, Finite State Transducers in Web Monitoring, Master Thesis, Faculty of Mathematics, University of Belgrade, Republic of Serbia (2010)

Appendix A

The results of monitoring the Blic web site from 9:00 PM 28.02.2012. until 9:00 PM 29.02.2012. (the unique concordances found by the WebMonitoring system):

plomatije Evropske unije Ketrin Ešton i	**predsednik Srbije** Boris Taidc izrazili s
ozo izjavio je danas posle razgovora sa	**predsednikom Srbije Borisom Tadicem** da s
Beogradu bio impresioniran posvecenošcu	**predsednika Srbije Borisa Tadica** evropsk
ki ispit koji položimo”, rekao je	**Tadic** i dodao da je status kandidata u s
u 241 Nikolic: Ovo nije primena zakona,	**Tadic** i DS vrše politicki progon protiv
e u sudnici i porucio da hoce da pobedi	**Tadica** i DS. Clan Predsednickog kolegiju
stervele razgovarace danas u Beogradu s	**predsednikom Srbije Borisom Tadicem** i mi
red optužbom radikala! PUPS bira izmedu	**Tadica** i Nikolica! SPO: Mrkonjic podržav
ic: Srbija zaslužuje status kandidata I	**predsednik Srbije Boris Tadic** i visoka p
Zahtevi Rumunije neopravdani Predsednik	**Tadic** izjavio je oko 13 casova da je oce
imo na pozitivnoj odluci - rekao je on.	**Tadic** je istakao da su zahtevi Rumunije
definišu kao rumunska manjina u Srbiji.	**Tadic** je izjavio i da ocekuje da ce evro
iterijumima", dodao je predsednik.	**Tadic** je porucio da Srbija ostaje privrž
za davanje statusa kandidata Srbiji, a	**Tadic** je precizirao da je Srbija ispunil
”, ali ostaje suzdržan optimista.	**Tadic** je rekao da smatra da je Srbija ob
prava", istakao je predsednik EK.	**Tadic** je rekao da Srbija ocekuje dalje n
voren centar NCR korporacije u Beogradu	**Predsednik Srbije Boris Tadic** je sa mini
biti status kandidata za clanstvo u EU.	**Tadic** je, izmedu dva sastanka Saveta min
Briselu predsednik Srbije Boris Tadic.	**Tadic** je, komentarišuci cinjenicu da se
Rumunije je ozbiljan problem U kabinetu	**predsednika Srbije Borisa Tadica** jutros
uaru. U Briselu ce danas i sutra biti i	**predsednik Srbije Boris Tadic** koji bi tr
Prištinu. Ona se zahvalila predsedniku	**Tadicu** na licnom angažovanju da podrži,
012. - 22:28h | Komentara: 5 Predsednik	**Tadic** na predavanju Pitera Bogdanovica P
se "smuca" po sudovima? Zašto	**Boris Tadic** nema probleme sa sudovima? N
Kandidatura nije i ulazak u EU Ešton i	**Tadic** Preporuka takode znaci, istakao je
kako je rekla, buducnost svoje zemlje.	**Predsednik Srbije** tom prilikom je podset
i ministri danas doneti odluku o Srbiji	**Predsednik Srbije** u Briselu je još jedno
protiv mene vrši Demokratska stranka i	**Boris Tadic** ", rekao je Nikolic. On
28.2.) biti odobren kandidatski status.	**Predsednik Srbije** , Boris Tadic, izrazio
tatusa još par dana? Ketrin Ešton je sa	**predsednikom Srbije**, Borisom Tadicem, ra
o na pozitivnom ishodu”, naveo je	**predsednik Srbije** , isticuci da ne &ldquo
kandidatski status. Predsednik Srbije,	**Boris Tadic** , izrazio je juce u Briselu n
eta runda dijaloga Beograda i Prištine.	**Predsednik Srbije Boris Tadic**, koji bora
m Manuelom Barozom. Ocekuje se da ce se	**Tadic** , nakon što ministri objave odluku
inim zemljama clanicama", rekao je	**Tadic** , osvrcuci se na zahteve Rumunije.
primamljiv osmeh, Mira Elizabet Fister	**Tadic** , pokušava da opiše sebe. U kratkim
apredenje obrazovanja." kaže se u ukazu	**predsednika Srbije Borisa Tadica**, povodo
ju na tom putu“, izjavio je Tadic. Nade	**predsednika Srbije**, pred današnju odluku
Ketrin Ešton je sa predsednikom Srbije,	**Borisom Tadicem** , razgovarala sinoc u Bri
o; Ostale i važne vesti: Povezane teme:	**Boris Tadic**, Savet ministara EU, Rumunij
Povezane teme: EU, Žoze Manuel Barozo,	**Boris Tadic**, Srbija, Kandidatura, Odluka
držimo Srbiju na tom putu“, izjavio je	**Tadic** . Nade predsednika Srbije, pred dan
je kosovsko pitanje”, istakao je	**Tadic** . On je ukazao da dobijanje statusa
ozo na zajednickoj pres konferenciji sa	**Tadicem** . Samit EU održava se 1. i 2. mar
pozitivnoj odluci, izjavio je u Briselu	**predsednik Srbije Boris Tadic**. Tadic je,
io je nemacki ministar. Povezane vesti:	**Tadic** : Nisam optimista kada je u pitanju
status kandidata uz napredak u dijalogu	**Tadic** : Postoji mogucnost da ne dobijemo
protestuju protiv Putina Vesti Politika	**Tadic** : Srbija ispunila uslove i za datum
asavanje Grcke: Politicari trce u krug »	**Tadic** : Srbija neće prihvatati nemoguće
sije RUBRIKE / Politika / Srbija Srbija	**Tadic** : Srbija zaslužuje status kandidata
oci pocetka Saveta za opšte poslove EU.	**Tadic** : Zahtevi Rumunije neopravdani Pred

Vesna Pajić is a teaching assistent at the Department of Agricultural Engineering, Faculty of Agriculture, University of Belgrade, Serbia, since 2003. She received Magister degree in Computer Science in 2010 and currently is doing her Ph.D. dissertation at the Computer Science Department of the Faculty of Mathematics, University of Belgrade. Her research interest includes natural language processing, computational linguistics, text mining, web search and bioinformatics.

Duško Vitas is a professor at the Department of Computing, Faculty of Mathematics, University of Belgrade, Serbia since 1994. Mr. Vitas received his Bachelor degree in Informatics in 1973, Magister degree in 1977, and Ph. D. degree in 1993, all in Mathematics at Faculty of Mathematics, Belgrade. Since 1991 he is employed at the Faculty of Mathematics, Belgrade. He published more than 120 scientific and professional papers.

Gordana Pavlović-Lažetić is a professor at the Computer Science Department of the Faculty of Mathematics, University of Belgrade, since 2009. She obtained her Ph.D. degree in 1988, at the Faculty of Mathematics, University of Belgrade. She spent two years at the University of California, Berkeley, doing research in database and text processing fields. Her current research interest includes databases, data mining, text processing and bioinformatics. She is an author of over 50 scientific papers and participated at more than 30 conferences. Professor Pavlovic-Lazetic supervised a number of Ph.D. and M.S. thesis. She participated in many national and international researches and was a member of organizing and program committees for several conferences. She is a member of ACM.

Miloš Pajić is a teaching assistent at the Department of Agricultural Engineering, Faculty of Agriculture, University of Belgrade, Serbia, since 2002. He received Ph.D. degree in Biotechnical Science in 2012 at the Faculty of Agriculture, University of Belgrade. His research interest includes technical inovations in biosistems, agricultural mechanization and bioinformatics.

Received: September 18, 2011; Accepted: July 18, 2012.

DOI:10.2298/CSIS1109260370

SLA-Driven Adaptive Monitoring of Distributed Applications for Performance Problem Localization

Dušan Okanović[1], André van Hoorn[2], Zora Konjović[1], and Milan Vidaković[1]

[1] Faculty of Technical Sciences, University of Novi Sad,
Trg D. Obradovića 6,
21000 Novi Sad, Serbia
{oki, ftn_zora, minja}@uns.ac.rs
[2] Software Engineering Group, University of Kiel,
Christian-Albrechts-Platz 4,
24098 Kiel, Germany
avh@informatik.uni-kiel.de

Abstract. Continuous monitoring of software systems under production workload provides valuable data about application runtime behavior and usage. An adaptive monitoring infrastructure allows controlling, for instance, the overhead as well as the granularity and quality of collected data at runtime. Focusing on application-level monitoring, this paper presents the DProf approach which allows changing the instrumentation of software operations in monitored distributed applications at runtime. It simulates the process human testers employ–monitoring only such parts of an application that cause problems. DProf uses performance objectives specified in service level agreements (SLAs), along with call tree information, to detect and localize problems in application performance. As a proof-of-concept, DProf was used for adaptive monitoring of a sample distributed application.

Keywords: continuous monitoring, adaptive monitoring, aspect-oriented programming, service level agreements.

1. Introduction

Modern enterprise applications constantly grow in size and complexity which makes them extremely demanding both from functional and non-functional aspects. Along with functional requirements, applications have to fulfill its non-functional requirements. Common non-functional requirements are availability, responsiveness, robustness, portability, etc. Non-functional requirements are defined in an agreement between software providers and consumers, called service level agreement (SLA) [1]. Before software is put into operation phase, in order to check software for bugs, it must be thoroughly tested. However, the testing phase of is often shortened, usually because of pressure to put the application in operation as soon as possible. Furthermore, the

standard testing, e.g., using debuggers and profilers, hardly allows detecting all errors and unpredicted events that occur in production or during operation. Also, it is a common phenomenon that software performance and quality of service (QoS) degrade over time [2]–which calls for continuous monitoring of applications in order to determine whether QoS is kept on a satisfactory level. Continuous monitoring of software is a technique that provides a picture of dynamic software behavior under real exploitation circumstances. The data obtained through the monitoring process can, for instance, be used as a basis for architecture-based software optimization, visualization, and reconstruction [3].

An important issue of software monitoring is imposed performance overhead, since the monitoring system shares common resources with the monitored system. Therefore, the monitoring system has to perform using a minimal amount of resources. In a testing phase, software developers commonly use tools such as profilers and debuggers. These tools induce significant performance overhead, and therefore, they are not suitable for monitoring during the operation phase. Monitoring code can only be optimized up to a certain extent. In order to achieve an even higher reduction of monitoring overhead, it would be beneficial to automatically adapt monitoring to only monitor selected parts of the system.

The DProf system proposed in this paper has been developed for adaptive monitoring of distributed enterprise applications with a low overhead. In order to do that, the Kieker [3] framework, which yields low overhead, is used for collecting the monitoring data. Additional components support changing of monitoring parameters at application runtime. These additional components have been developed using Java Management Extensions (JMX) [4]. The system analyzes call trees (as described in the following paragraph) reconstructed from the gathered data and automatically creates a new monitoring configuration if needed.

A call tree represents calling relationships between software methods [5]. It contains the control-flow of method executions invoked by a client request. The first method is called the "root". For example, consider the simplified call tree in Fig. 1. This call tree represents a situation where a client invokes *methodA()* from *ClassA*. This method in turn, invokes two methods from *ClassB*: first *methodB1()* and then *methodB2()*. *SRVX* and *SRVY* are the names of servers on which the methods are being executed.

DProf configuration parameters specify which of the application's call trees are going to be monitored and, furthermore, they can specify nesting levels within the call tree that are to be monitored. DProf stores data in a central database, regardless of on how many computers the monitored application is executed. Using mechanism integrated into the Kieker framework, during data gathering, each method execution within a trace is uniquely identified and assigned a number which represents the order of execution (numbers on branches in Fig. 1). This allows call trees to be spread on different computers.

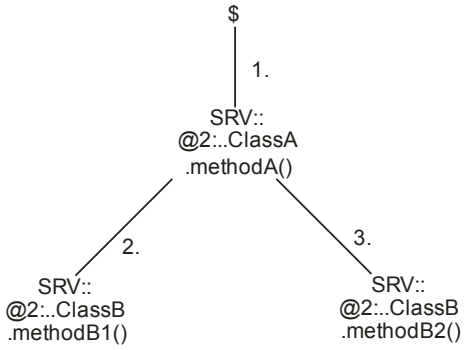


Fig. 1. An example call tree

DProf can be configured to work in different modes, e.g., for the following purposes: 1) locating software components causing deviations between obtained results and values required by service level agreements (SLAs), 2) detecting bottlenecks, or 3) collecting performance data for post-mortem analysis. The first two modes are usually used for problem detection and localization, while the third mode is used when software performance ought to be evaluated in general. DProf uses SLAs that are defined in an XML document, for which we propose an XML schema, called DProfSLA. The schema is compliant to existing SLA standards in the field.

The idea behind our approach is to reduce monitoring overhead by only monitoring parts of software suspected of containing problems or deviating from expected behavior. In the problem localization process, the system starts by monitoring methods that are at the root of call trees. If the deviation from expected results in one of the trees is detected, the DProf incrementally turns on monitoring in lower levels of that particular tree. This is repeated successively, until the method that is causing the problem is determined. DProf adapts without human intervention to find the cause of the problem.

This simulates the manual procedure typically employed for localizing the root cause of performance problems. Other systems perform monitor the whole software, regardless of the fact that other parts (other call trees) are working fine. Since DProf's additional monitoring components are implemented using JMX technology, reconfiguration of the DProf monitoring parameters can still be performed manually by system administrators using any JMX console.

Software administrator intervention is only needed at the beginning of the monitoring process, when the monitoring goals are configured. It usually takes some time before clients start reporting a performance problem and even more until the service provider reacts, locates the problem, and finds a solution. Automation of localizing performance problems and faults reduces this time. DProf can detect even the slightest deviations proactively. This can provide enough time to react before clients start complaining, leaving software performance at desired levels.

In our earlier work we presented some parts of the monitoring subsystem of the DProf system [6, 7]. In this paper, we further extend those results with automatic adaptation of the monitoring process. We presented the DProfSLA XML schema in [8]. This paper presents an enhanced version of the schema, which contains support for the latest DProf features. A more detailed evaluation of the system is also presented.

The remainder of the paper is organized as follows. In Section 2 we present the DProf monitoring system, including its components, architecture and functions provided. Section 3 presents an evaluation of the DProf monitoring system. Section 4 discusses related work. It contains an example of the continuous and adaptive monitoring of a real application and presents a discussion of the obtained monitoring results. Finally, Section 5 draws concluding remarks and outlines directions for future work.

2. DProf System

The DProf system enables adaptive monitoring of distributed enterprise applications with a low overhead. It performs automatic analysis of obtained data based on call tree analysis and automatically reconfigures the monitoring instrumentation in order to reduce performance overhead or to provide more detailed data. The system configuration specifies which parts of the application are going to be monitored by selecting an application's call trees and levels within these call trees.

DProf is based on the Kieker framework and the JMX technology. It can be used for adaptive and reconfigurable continuous monitoring of Java EE applications, as presented in this paper. Use of Kieker grants low overhead. Separation of monitoring code from application code and source code instrumentation is performed by using aspect-oriented programming (AOP) [9]. We have developed additional components in order to allow an adaptive reconfiguration of monitoring parameters at runtime, i.e., while the application is running. JMX is used for controlling the monitoring process at runtime. Together with the DProfSLA schema, DProf can be used to monitor SLAs compliance and to localize the root cause of problems.

Details of our approach are presented in Section 2.1. In Section 2.2 we describe the DProfSLA XML schema. An overview of the underlying Kieker framework is given in Section 2.3. Section 2.4 presents architecture and some implementation details of the DProf system.

2.1. The DProf Approach

The activity diagram in Fig. 2 illustrates the DProf monitoring process. Before the application is started, an initial monitoring configuration is specified using *include* and *exclude* clauses in the *aop.xml* file, which configures the AOP-based instrumentation.

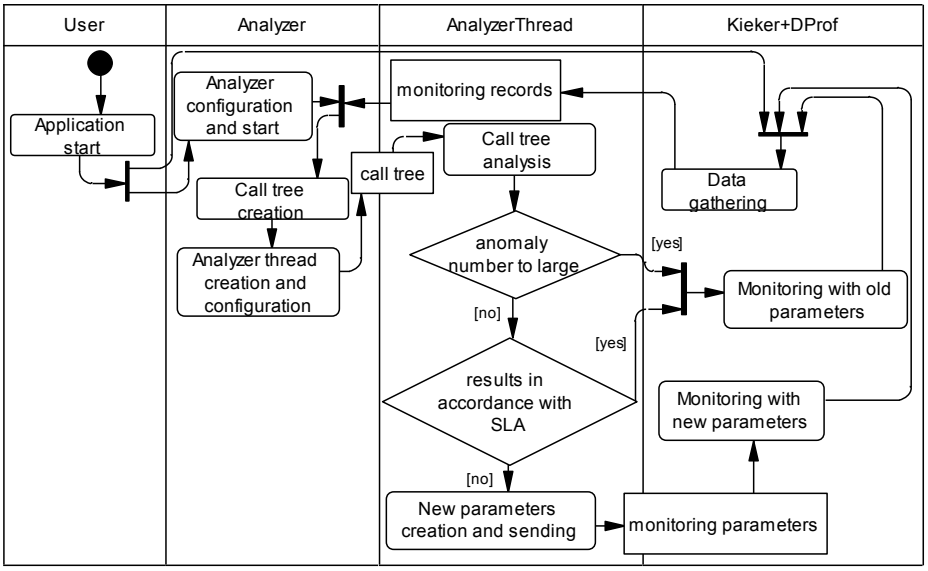


Fig. 2. Activity diagram of the DProf monitoring system

On application startup, with the initial monitoring parameters specified in the *aop.xml*, the DProf system is started simultaneously. It gathers monitoring data during application execution. Periodically, obtained performance data is being sent for analysis. The *Analyzer* reconstructs call trees based on monitoring data. These trees are analyzed by the *AnalyzerThreads*, each thread analyzing one tree in parallel to speed up analysis. A call tree represents methods that are invoked after one client call to the application. Each method invocation in the stack trace is represented with one node of the tree.

For the analysis we use the R [10] programming language and environment for statistical computing. We use the *extremevalues* [11] package to detect and remove outliers that we consider temporary effects caused by various external factors: class loading, starting of some resource-consuming process in the background while the monitored application is running, hardware glitches, etc. After outlier removal, the remaining values are processed using the specified statistical function and compared to the required value as defined in the SLA. Depending on the result of the comparison, new monitoring parameters are generated. If the number of outliers exceeds the value defined in DProfSLA, monitoring is repeated with old parameters.

If results deviate from values defined in the SLAs, the *AnalyzerThread* creates new monitoring parameters. The creation of new parameters depends on monitoring configurations defined in the SLA document. The system can be configured to monitor all or only selected parts of the application for the following purposes:

1. Recording normal results – this is used to determine nominal values for SLAs. No changes in monitoring parameters are assumed in this case.
2. Finding which software component does not conform to the SLAs – in the SLAs we provide nominal values for nodes in call trees we want to be monitored.
3. Finding which software component consumes the largest amount of resources.

Using the DProf system, developers cannot only find which method causes problems, but also in which context the problems occur. Since the communication between the Analyzer and the components that are gathering the data is implemented using web services, this component can be used for receiving and analyzing monitoring records from applications developed for platforms other than Java/Java EE. In order to use this system with some other platform, such as .NET, adapters for the monitoring subsystem and the management interface are required.

SLA Compliance Monitoring and Problem Localization

In order to provide desired values for SLA, the application is monitored using the first configuration from the previous section (recording of normal results). Branches omitted from the SLAs are not monitored.

DProf starts with monitoring the top levels specified. If a problem is detected in one of the call trees, DProf triggers a reconfiguration to include monitoring of the next level of that tree. It will proceed down the tree as long as there is a discord with SLAs. The last node with values higher than those in SLA is declared the source of the problem.

Localization of Increased Resource Consumption

In the DProfSLA document we specify which call trees are to be monitored. For each call tree, the Analyzer configures the monitoring system to gather data only from the top level. In the next iteration, it finds the tree with the highest observed value (that is a root element of that tree). In the next iteration, the monitoring system is reconfigured to monitor only that call tree's first two levels. This process is repeated further down the tree (if those levels exist). Through the process, DProf selects the branches with the highest observed values. The process ends as soon as the instrumentation reaches the bottom of the call-tree, or when observed values for the node on the higher level are greater than the values for its child nodes.

2.2. DProfSLA Schema

DProfSLA documents are used to define SLA parameters based on our DProfSLA XML schema. The relevant part of this schema with the root element and its sub elements of this schema is shown in Fig. 3. (In this paper we use the XMLSpy [12] notation for the XML schema representation.) The root element (*DProfSLA*) has three sub elements: *Parties* (parties in the

agreement), *CallTreeNode* (call-traces to be monitored) and *Timing* (time constraints of this agreement).

The *Parties* element represents the parties involved in the agreement. This element has two sub elements: service provider (*Provider)* and service consumer (*Consumer*). Both of these sub elements contain contact data regarding the service provider and service consumer respectively. Each sub element is represented using the *OrganizationType* complex type (not detailed here).

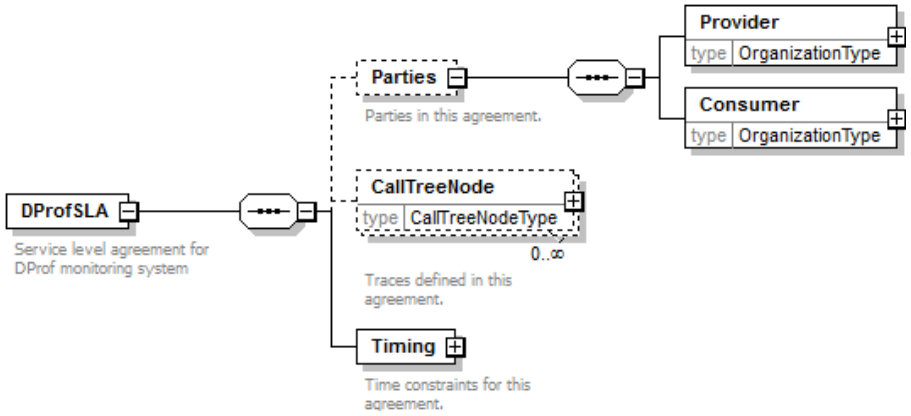


Fig. 3. A part of the DProfSLA schema with the root element

Selection of Call Trees to be Monitored

Each *CallTreeNode* element represents performance information for a single node in the call tree to be monitored. It is of the *CallTreeNodeType* complex type shown in Fig. 4.

CallTreeNodeType elements have two mandatory attributes, a *name* and a *metric*. The *name* attribute is used to specify a part of the application to be monitored. The string representation of a call tree is used for this purpose. The metric attribute specifies the performance metric to be used, i.e., which aspect of application performance is going to be monitored (e.g., response time, memory consumption). Sub elements of this element are other sub call trees, e.g., sub traces that are invoked from the parent *CallTreeNode* element.

Furthermore, optional attributes for specifying expected performance values in terms of the designated metric can be configured. The *aggregateFunction* represents the function to be used in data analysis. The *nominalValue* represents the expected value (for the given aggregate function), while the *upperThreshold* and the *lowerTreshold* are maximal and minimal values of the designated metric, respectively. The *outlierPct* is used to define the allowed fraction of outliers (Section 2.1) in the set of obtained results.

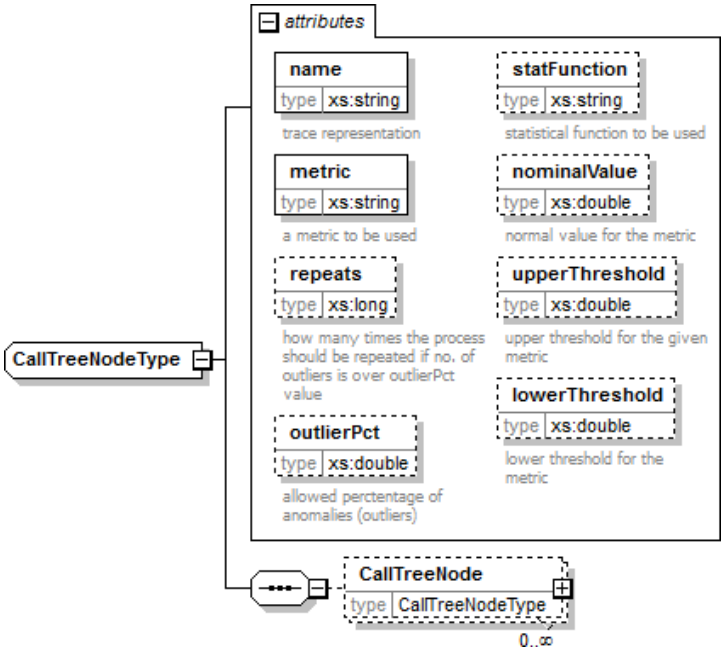


Fig. 4. *CallTreeNodeType* complex type defined in the DProfSLA schema

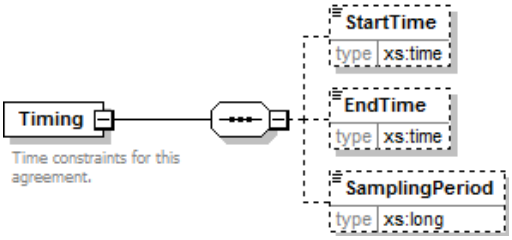


Fig. 5. *Timing* sub element in the DProfSLA schema

Specification of Timing Constraints

The *Timing* element (Fig. 5) is used to specify time constraints for this agreement. The sub elements *StartTime* and *EndTime* define the period this document applies to. The *SamplingPeriod* element denotes the time period (in milliseconds) between two analyses runs, possibly resulting in a reconfiguration of monitoring parameters.

Example DProfSLA Document

An example DProfSLA document, which describes monitoring of the call tree from Fig. 1, is shown in Listing 1.

```
1  <DProfSLA>
2   <Parties><Provider name="Org1" />
3      <Consumer name="Org2" /></Parties>
4   <CallTreeNode metric="avgExecutionTime"
5         name="ClassA.methodA,[{ ClassB.methodB1,[]} ,
6           {ClassB.methodB2,[]}]" upperThreshold="350">
7      <CallTreeNode metric="avgExecutionTime" name="[{
8         ClassB.methodB1,[]}]"  upperThreshold="150"/>
9      <CallTreeNode metric="avgExecutionTime" name="[{
10        ClassB.methodB2,[]}]"  upperThreshold="150"/>
11  </CallTreeNode>
12    <Timing><SamplingPeriod>600000</SamplingPeriod></Timing>
13 </DProfSLA>
```

Listing 1. DProfSLA document for this example

It represents an agreement between the parties Org1 and Org2. Response times are monitored to detect values exceeding the specified *upperThreshold* attribute. Every 10 minutes (600,000 ms), an analysis of the obtained results is performed.

In the first iteration the system only monitors *monitorA()*. If the obtained results show that the response times of *methodA()* exceed the upper threshold, monitoring of *methodB1()* and *methodB2()* is turned on. After the next 10 minutes, if results show that either *methodB1()* or *methodB2()* takes too long, it will have to be analyzed manually. Otherwise, the program code in *methodA()* is assumed to be the cause of the problem.

2.3. Kieker Framework

The Kieker framework is structured into the *Kieker.Monitoring* and the *Kieker.Analysis* components [3]. The *Kieker.Monitoring* component collects and stores monitoring data. The *Kieker.Analysis* component performs analysis and visualization of the monitoring data. The core components of the Kieker framework are depicted in Fig. 6, and described in the remainder of this section.

The *Kieker.Monitoring* component is executed on the same computer the monitored application executes on. This component collects application-level measurement data during the execution of the monitored applications. *Monitoring Probes* are software sensors that are inserted into the monitored application in order to gather various measurements. For example, Kieker includes probes to monitor control-flow and timing information of method executions. Probes are most commonly implemented using AOP technology; additional probes can be added to support different measurements, e.g., for adding support for new metrics. *Monitoring Writers* pass the collected data (as *Monitoring Records*), to a *Monitoring Log or Stream*. The framework is distributed with *Monitoring Writers* that can store *Monitoring Records* in, for example, file systems, databases, or Java Message Service (JMS) queues

[13]. Additionally, users can implement and use their own writers, as we did for DProf. The *Monitoring Controller* component controls the work of this part of the framework.

The data in the *Monitoring Log/Stream* is analyzed by the *Kieker.Analysis* component. A *Monitoring Reader* reads records from the *Monitoring Log/Stream* and forwards them to a pipe-and-filter configuration of *Analysis Filters*. *Filters* may, for example, analyze and visualize gathered data. Control of all components in this part of the Kieker framework is performed by the *Analysis Controller* component.

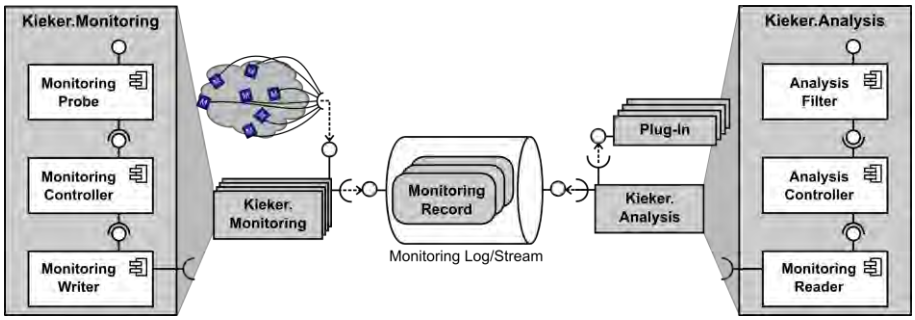


Fig. 6. Component diagram of the Kieker monitoring framework

2.4. DProf System Architecture

We have implemented our approach using Java technology. The DProf system uses Kieker's infrastructure for data acquisition, extended by some additional components. The architecture of DProf system and its integration with Kieker are shown in Fig. 7.

The DProf components are divided into two groups: i) components that participate in recording monitoring data; and ii) components that analyze the obtained data and control the reconfiguration of monitoring parameters.

The *DProfWriter* is the new *Monitoring Writer* used. It sends *Monitoring Records* to the *ResultBuffer* component. The *ResultBuffer* periodically sends data to the *RecordReceiver* component, which, in turn, stores data into the relational database. The combination of *ResultBuffer*, *RecordReceiver,* and database plays the role of the *Monitoring Log/Stream* (Section 2.3).

Received data is periodically analyzed by the *Analyzer* component. The *Analyzer* is responsible for controlling the monitoring configuration. Configuration parameters are sent to the *DProfManager* component, which passes these parameters to the *AspectController* and to the *ResultBuffer* (to clear, if it contains result created with previous configuration parameters). The *AspectController* accesses the application's *aop.xml* file and performs changes, causing the application to restart. Upon the restart the new monitoring parameters are applied.

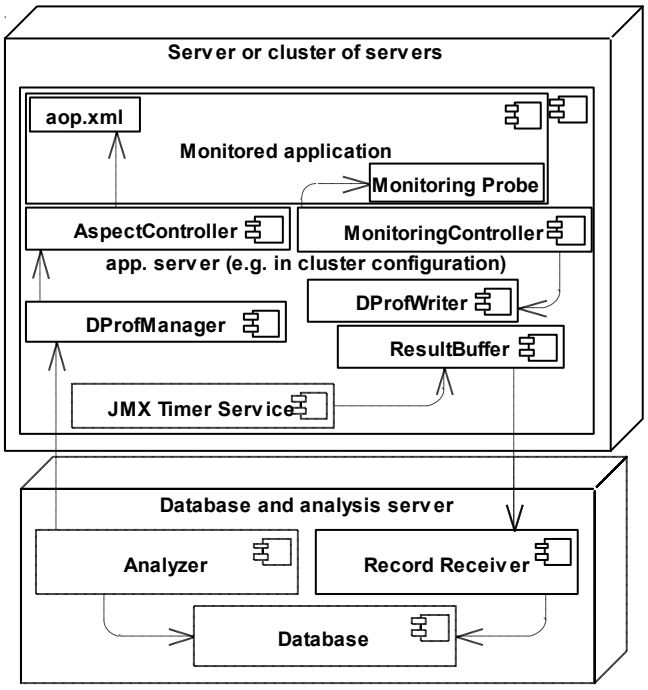


Fig. 7. Deployment diagram of the DProf system

Kieker includes the monitoring record type OperationExecutionRecord that is used to store timing and trace information for method executions. We have developed the new *Monitoring Record* type *DProfMonitoringRecord*, which extends Kieker's original *OperationExecutionRecord* and additionally provides the *otherData* attribute. This attribute is used to store additional information, e.g. CPU utilization and memory consumption. When the record is created in the probe, the attribute is filled with comma-separated key-value pairs, depending on what the given monitoring aspect measures. Keys in this list correspond to metrics defined in the SLA document. This allows us to use this single Monitoring Record class for monitoring different metrics.

The *RecordReceiver* receives the data from the *ResultBuffer*. It is implemented as a web-service, and it stores records into a database table.

By using the *DProfManager* and these additional components we can change monitoring parameters at runtime. This allows us to reduce the impact on the system, including monitoring overhead, by disabling monitoring in certain parts of the application, and to obtain more accurate results. Setting the new parameters can be performed either manually, by a person in charge or automatically by the *Analyzer* component. The *Analyzer* component, provided with a DProfSLA schema document, can check if service levels observed in gathered data deviate from those defined in the SLA and, according to the algorithm described in Section 2.1, to determine which part of the software causes this deviation.

Code instrumentation can be performed by hard-coding instrumentation routines into program code, but a more elegant way is AOP . AOP provides developers with separation of concerns: monitoring aspects are developed separately from application code.

Using AOP, we can choose to weave aspects with code upon compilation or to let the aspect runtime weave aspects into classes upon class loading. These processes are known as compile-time-weaving and load-time-weaving. When using DProf, we usually want to change monitoring parameters at runtime, so we use load-time-weaving. If we monitor, without having to change monitoring parameters at runtime, we can use compile-time weaving. The advantage of using compile-time-weaving is only a faster application start; afterwards both compile- and load-time-weaved applications behave the same.

The DProf system uses the AspectJ AOP implementation for Java [14], for instrumentation. Initially, the AspectJ configuration file (*aop.xml*) specifies which parts of the application are to be included/excluded from monitoring, and which aspect to use as monitoring probes. During monitoring with the DProf system, additional clauses will be placed in this configuration file for the purpose of monitoring adaptation.

In the Java environment, time is usually measured using either *System.currentTimeMillis()* or *System.nanoTime()* calls [15]. Measuring of system-level metrics (such as memory consumption and CPU utilization), can be performed using platform MXBeans [4] or some third-party tools such asSIGAR [16].

3. Evaluation of the DProf System

The application of the DProf system will be demonstrated using the software configuration management (SCM) application described in our previous work [17]. SCM is a Java EE application responsible for tracking of applications and application versions in a company.

The goal is to monitor method response times and to localize the root cause of performance problems. Initially, DProf is configured to monitor only methods at the root of call trees. If an increase in method response times is detected, DProf will, potentially successively, reconfigure the instrumentation to monitor other levels, until it localizes the method that causes the problem.

This evaluation serves to demonstrate that monitoring overhead can be reduced by monitoring only root level if no performance problem is present. Also we perform a basic analysis of the overhead generated when using DProf, comparing it to the overhead generated by writers distributed with the Kieker framework.

3.1. Setting

The application is implemented using Enterprise JavaBeans (EJB) [18] technology. Entity EJBs are used for communication with databases, i.e., for object/relational (O/R) mapping [19]. They are accessed through stateless session EJBs (SLSB), modeled according to the façade design pattern [20]. SLSBs are annotated to work as JAX-WS [21] web services as well. We deployed SCM on a cluster of servers. The application client is a Java Swing [22] application.

Figure 8. shows a part of the application's architecture.

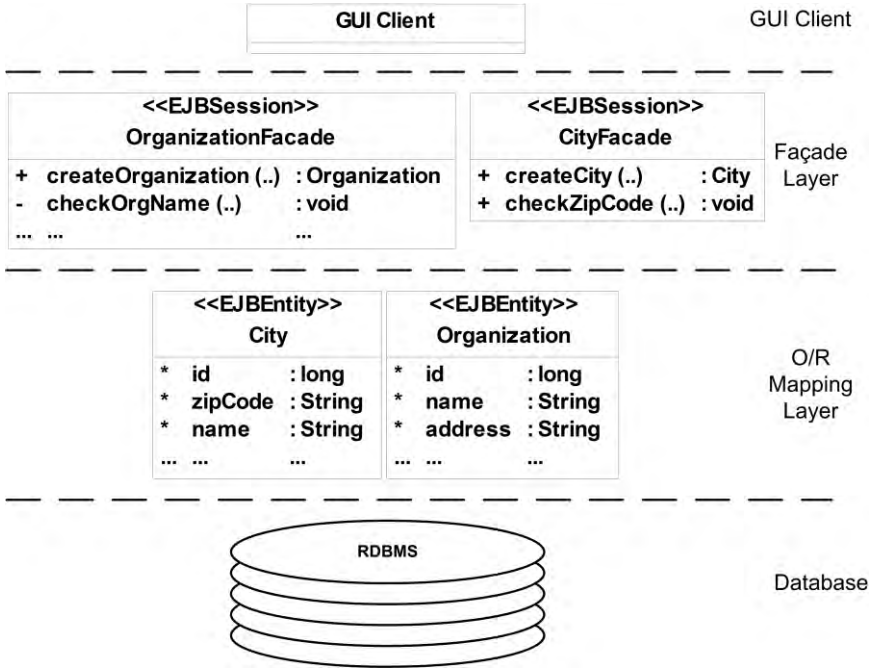


Fig. 8. A part of the monitored SCM application's architecture

Methods that are to be monitored are annotated with Kieker's *@OperationExecutionMonitoringProbe*. As a monitoring probe we used a Kieker's original *OperationExecutionAspectAnnotation* probe. It intercepts executions of annotated methods.

In this case study we will focus on the call tree shown in Fig. 9.

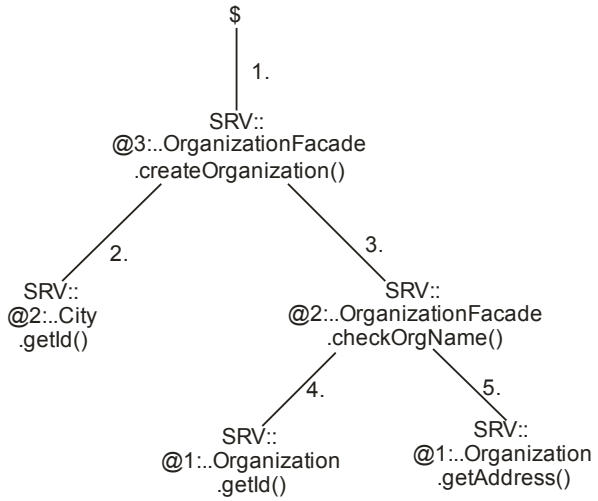


Fig. 9. The call tree monitored in this example

The testing was performed by repeatedly invoking the *OrganizationFacade.createOrganization()* method from 100 concurrent threads, with equally distributed think times between 0 and 10 seconds.

The analysis of the obtained data is performed every hour. Initially, only the *createOrganization(..)* method is monitored. After a deviation from values specified in the DProfSLA (last row in Table 1.) is detected, the methods invoked from this one are monitored additionally. If these methods do not violate the SLAs, the problem is assumed to be in the *createOrganization(..)* method. If the results for the *checkOrgName(..)* show deviations, monitoring is reconfigured to include the *Organization.getId()* and *Organization.getAddress()* methods, and to exclude the method *City.getId()*. The most likely cause of the problem is the method whose results do show deviation from expected response times, while methods invoked from it do not.

Within the *checkOrgName()* method, we purposly inserted a delay of 1 ms, to simulate a problem. In order to determine the impact of DProf on the monitored application, we measured response times on the client computer.

3.2. Analysis of Results

The obtained results were analyzed by the *Analyzer* after one hour, showing increased response of the *createOrganization(..)* method.To find the source of the problem, the *Analyzer* component changed monitoring parameters and added monitoring instrumentation to the methods in the next level of the call tree.

The analysis of the gathered data, one hour after the previous analysis, showed that an response time of the *checkOrgName(..)* method rose over designated values. The *Analyzer* then included the monitoring in the third level, i.e., the methods *Organization.getId()* and *Organization.getAddress()*. The obtained results are shown in Table 1.

Table 1. The average response times of monitored methods in milliseconds

Method / *Levels monitored*	Organization-Facade. create-Organization	City. getId	Organization-Facade. checkName	Organization. getId	Organization. getAddress
1	2.888	Not monitored	Not monitored	Not monitored	Not monitored
1 and 2	3.05	0.307	1.502	Not monitored	Not monitored
1, 2 and 3	3.339	Not monitored	2.290	0.429	0.71
Response times required by the SLA	2.250	0.750	1.300	0.750	0.850

Organization.createOrganization(..) has increased response time because of the *OrganizationFacade.checkOrgName(..)*. In turn, increased results of *OrganizationFacade.checkOrgName(..)* are not caused by the executions of the *Organization.getId(...)* and *Organization.getAddress(...)* methods.

Based on these results, it can be concluded that the *checkOrgName(...)* method requires further inspection in order to be made compliant in accordance to the SLA. This means that our system has been able to localize the method which causes the problem.

Overhead analysis

In order to estimate overhead we measured response times on the client side. A comparison of these times is shown in Fig. 10. The median response time of the monitored method, when monitoring is disabled, was 3.078 ms. By enabling monitoring of the call tree's first level, it increased to 3.535 ms. Monitoring of the second level generated additional 0.344 ms (it increased to 3.879 ms). Inclusion of monitoring of the third level led to average response time of 4.133 ms.

As expected, DProf yields an overhead, which rises if we increase the number of monitored methods. Also, a slight increase of the standard deviation in results from 0.954 ms to 1.194 ms shows that reponsiveness becomes more unstable when the number of monitored call tree levels is increased. Hence, in case no problem is detected, the overhead would be minimal and responsiveness more stable, since only the first level would be monitored.

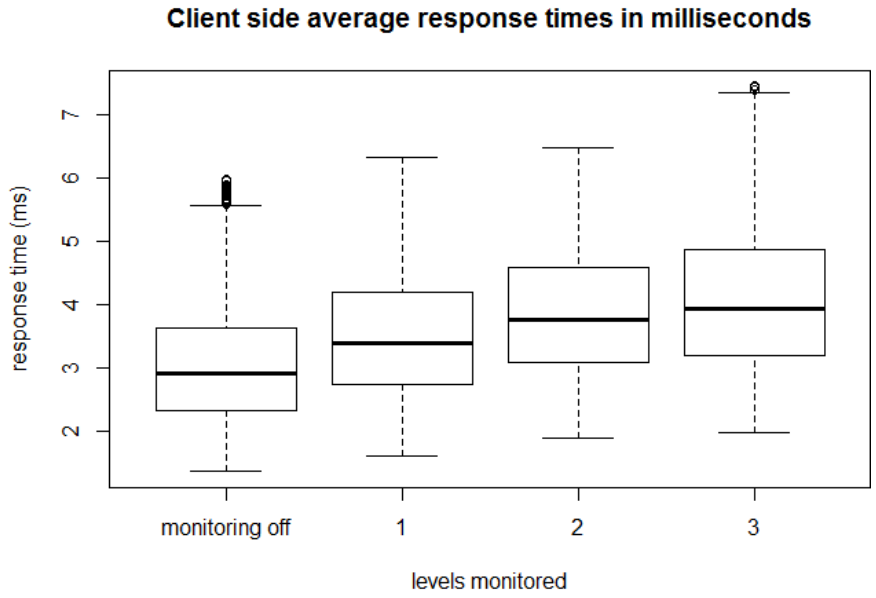


Fig. 10. Comparison of response times of the *Organisation.createOrganisation(…)* method in different scenarios

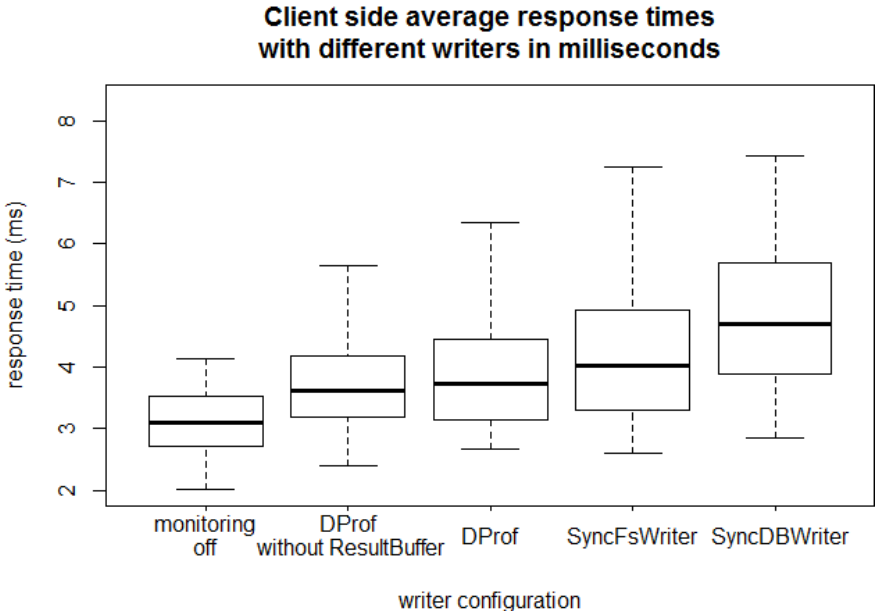


Fig. 11. Overhead comparison for different writers

The obtained results are also in accordance to Kieker overhead analysis shown in [3]. Further comparison with Kieker's writers is shown in Fig. 11. We compared response times of the monitored application in different monitoring configurations: with no monitoring, when using DProf with and without sending data to the ResultBuffer, and when Kieker's original *SyncFSWriter* and *SyncDBWriter* are used.

The DProf system has lower overhead than Kieker's original *SyncFSWriter* and *SyncDBWriter,* which write records into file system and database, respectively. This is because in DProf, communication between the writer and the buffer is performed within one JVM.

Based on these results, it can be concluded that this system is suitable for continuous monitoring of all kinds Java applications. It provides valuable data on application execution with very small impact on application performance.

4. Related Work

For the research presented in this paper two fields are of particular importance: monitoring tools (which are presented in Section 4.1) and existing standards for SLA documents definition (Section 4.2).

4.1. Application Monitoring and Profiling Tools

Monitoring and profiling tools have been in use since the early 1970s. The UNIX operating system includes the *prof* tool [23] since 1979. It can record execution times for each program function. ATOM [24] was one of the first to use source code instrumentation and it appeared in the 1990s. Before application deployment, ATOM combines the instrumentation and the application code. The application executes normally, with additional output containing monitoring data.

A recent study by Snatzke [25] shows that, although service levels and performance of applications are of critical importance in practice, application level monitoring tools are rarely used. Java application monitoring tools are usually developed using either JVMTI/JVMPI [26, 27] or aspect-oriented programming (AOP) [8] technology. JVMTI and JVMPI APIs require knowledge of C/C++ in addition to Java, and also impose significant overhead [3]. Examples of JVMTI/JVMPI-based profilers are JBoss Profiler [28] and JFluid [29]. JBoss Profiler is the profiler used with the JBoss application server [30]. JFluid is used within the NetBeans IDE [31]. COMPASS JEEM [32] can be used to monitor JEE applications, but every application layer needs a different set of probes. The Kieker framework [3], used in this work, is a Java-based framework for continuous application performance monitoring and dynamic software analysis. It includes aspects which implement monitoring probes.

A number of commercial application monitoring tools exist, but implementation details of these tools are scarce at best, if available at all. DynaTrace [33] uses its own PurePath technology which captures timing and context information for transactions across all application tiers. It has support for both Java and .NET environments. JXInsight [34] is designed for monitoring applications in JEE environments. It offers automatic performance analysis and problem detection. IBM's Tivoli Management Framework [35] is a system management platform. It is CORBA-based and allows remote management of software. IBM Tivoli Monitoring, which uses the Tivoli framework, is a set of tools which can be used for problem detection in various environments. Tivoli supports monitoring of JavaEE (WebSphere server), .NET, network (DNS, DHCP) and others. Both agent and agentless monitoring are supported. AppDynamics provides solutions for monitoring on different platforms, with low overhead [36]. It supports the automatic localization of problem root causes. Monitoring tools for other purposes exist as well, e.g., Nagios [37] for infrastructure monitoring, CA Unicenter [38] for infrastructure and application performance monitoring and management, or HP's Insight [39] for monitoring and problem localization on some specific platforms.

Newman et al. present the MonALISA system [40] which constitutes a distributed monitoring service. It is implemented using Java and WSDL/SOAP technologies. MonALISA allows for monitoring of heterogeneous systems using autonomous agent based sub-systems. A graphical user interface visualizes complex gathered data. MonALISA includes a library of APIs that can be used to send data to MonALISA services. Using these APIs, other systems, such as DProf can be included in the monitoring process.

AOP can be used for instrumentation of code. Separation of concerns allows for monitoring code to be separated from application code. There are several monitoring tools based on AOP.

The concept of manageable aspects–a combination of aspects and JMX MBeans–is proposed by Liu et al. [41]. It can be used as monitoring probes, for instrumentation and collecting runtime data during software execution. They can be accessed and controlled using any JMX console. Although this approach would present an excellent platform for adaptive monitoring, no implementation of this concept has been provided, yet.

The HotWave framework [42], which is still in development phase, allows run-time reweaving of aspects and the creation of adaptive monitoring scenarios. It allows for a development of adaptable monitoring solutions, as presented by the authors. Users can choose parts of the application to be monitored, and later reconfigure the system to monitor other parts, without having to restart the system. Unfortunately, no implementation of this framework is currently available.

Ehlers et al. present an approach for anomaly diagnosis [43] also based on call tree analysis and self-adaptive monitoring with Kieker. For each call tree node, representing the execution of a software method in a certain context, anomaly scores for response times are computed by comparing observed values with values predicted based on historic observations. OCL [44] is used

to specify rules for adapting the instrumentation based on the anomaly scores and the current instrumentation. In our earlier work [45], we presented an approach for automatic problem localization based on a correlation of anomaly scores with architectural calling dependencies. Kieker was also used in this approach. However, the monitoring was not adaptive.

Yu et al. [46] present the RaceTrack tool for race detection in .NET applications. This tool monitors program activity and looks for suspicious patterns in program execution. It has great accuracy because it monitors memory access at both object and field level. It starts by monitoring at object level, and only if unusual patterns are detected, it switches to field level. This way, performance overhead is reduced. The RaceTrack is implemented by modifying .NET's virtual machine CLR (*common language runtime*). Such modification requires great understanding of how CLR works. If some changes are made in the future, it would probably require modifications on this tool. Also, the modified CLR has to be distributed with the application that is to be monitored, instead of, for example, just starting a tool within existing CLR.

Chen et al. [47] propose the Pinpoint system that locates components most likely to cause a fault. The approach is based on finding correlations between low-level faults and high-level problems. Data is gathered by collecting client traces using a modified Java EE platform. Unlike our approach, this approach focuses more on problem localization and less on performance problems.

A black box approach to problem localization is applied by some of the authors. This approach usually finds a component that is causing problems, but does not locate the problem within component. Aguilera et al. [48] use an approach that monitors message communication between components and tries to find causal paths between messages and performance problems. The PeerPressure tool presented by Wang et al. [49] compares "healthy" and "suspicious" machines using statistical methods to locate problems.

Very few papers provide actual numbers regarding overhead. Dimitriev [29] tested JFluid's performance with *SPECjvm98* tool [50]. Results show that overhead ranges from 1% for time consuming tasks like database access, to 5000% for *compress* tasks. JFluid allows users to reduce overhead by selecting the parts of an application to monitor. Govindraj et al. [51] discuss a possibility of using AOP for monitoring and they show the overhead ranges from 1-10%. For DynaTrace the monitoring overhead is reported to be less than 5%. However, these percentages are hardly comparable because they heavily depend on hardware and software used in the benchmarks, and especially they depend on the granularity of instrumentation and the usage profile.

4.2. SLA Standards

In order to automate service level management, SLAs must be defined in machine-readable format. As shown by Tebbani et al. [52], only few formal SLA specification languages exist. In practice, SLAs are often written in some

informal language. Tebbani et al. propose the GSLA (Generalized Service Level Agreement) language. A GSLA document constitutes a contract signed between two or more parties designed to create a measurable common understanding of each party's role. The role is nothing but the set of rules which defines the minimal service level expectations and obligations the party has. GXLA is the XML schema which implements the GSLA information model. GXLA documents are composed of the following sections: schedule (temporal parameters of the contract), party (models involved parties), service package (an abstraction used to describe services) and role (as described). The use of GXLA supports an automation of the service management process.

WSLA [53] is a language to specify service levels for web services. XML-based WSLA documents define the involved parties, metrics, measuring techniques, responsibilities, and courses of action. The authors state that every SLA language, such as WSLA, should support contain 1) information regarding the agreeing parties and their roles, 2) SLA parameters and a measurement specification as well as 3) obligations for each party.

SLAng [54] is a language for specifying SLAs based on the Meta Object Facility [55]. It can use different languages for describing constraints, e.g., utilizing OCL [44] or HUTN [56].

The WS-Agreement specification language [57] has been approved by the Open Grid Forum. It defines a language for service providers to offer capabilities and resources, and clients to create an agreement with that provider.

Paschke et al. [58] propose a categorization scheme for SLA metrics with the goal to support the design and implementation of SLAs that can be monitored and enforced automatically. Standard elements of each SLA are categorized as: technical (service descriptions, service objects, metrics, and actions), organizational (roles, monitoring parameters, reporting, and change management), and legal (legal obligations, payment, additional rights, etc.). Paschke et al. categorized service metrics in accordance with standard IT objects: hardware, software, network, storage, and help desk. SLAs are grouped according to their intended purpose, scope of application, or versatility.

According to this categorization, DProfSLA documents (described in Section 2.2) are operation-level documents intended to be used in-house. By versatility categorization, they belong to standard agreements. We chose to design our own XML schema as an intermediate format, because we do not need all of the features of the described schemas. It is specifically designed to be used with the DProf system. Our schema provides a subset of the elements defined by GXLA or WSLA. A transformation of SLA documents between DProfSLA and the mentioned schemas could, for example, be performed using XSLT.

5. Conclusion

This paper presented the DProf approach for continuous and adaptive monitoring of distributed software systems and automatic evaluation of software performance against expected values defined in service level agreements (SLAs). The DProf system gathers data from application execution, compares these measurements with the SLAs and, based on call tree analysis, aims it localizes application components causing possible SLA violations. Expected values are defined in a document based on the described DProfSLA XML schema. The schema is designed with existing SLA schemas, such as GXLA and WSLA, and their categorizations of contained information in mind. DProfSLA's intended use is for standard intra-organizational agreements, but it may be used for inter-organizational agreements, too. The schema supports various metrics and additional metrics can be added as needed.

The DProf monitoring system is mainly designed for continuous monitoring of JEE applications, but with minor modifications it can be used to monitor applications developed for other platforms. We described the architecture of our DProf prototype, whose implementation is based on the Kieker framework with additional JMX-based components.

As a proof-of-concept, the DProf system was used for adaptive monitoring of a sample Java EE application. The analysis of obtained results shows low monitoring overhead, and reduced overhead by enabling monitoring on-demand.

Our system is not able to differentiate between call trees with the same root element, that can have different lower nodes. In this case the system could report incorrect results. In order to confront this issue, developers should choose to monitor only one of these trees, and exclude the other using an appropriate *aop.xml* configuration file.

Our future work regarding DProf will focus on the implementation of the DProf Analyzer as a Kieker plugin and an integration of the DProf component into the Kieker distribution. We also plan to further extend the system by additional monitoring probes for different and more complex measures. Furthermore, we will work on more advanced algorithms for the Analyzer component, enabling it to change monitoring parameters on different computers in distributed environments.

References

1. Benyon, R.: Service Agreements: A Management Guide. Van Haren Publishing, Netherlands. (2006)
2. Grottke, M., Matias Jr., R., Trivedi, K. S.: The Fundamentals of Software Aging. In Proceedings of the 1st International Workshop of Software Aging and Rejuvenation/19th International Symposium on Software Reliability Engineering (WoSAR/ISSRE). Seattle, USA, 1-6. (2008).

3. Hoorn, A. v., Hasselbring, W., Waller, J.: Kieker: A Framework for Application Performance Monitoring and Dynamic Software Analysis. Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering (ICPE 2012). ACM, Boston, Massachusetts, USA. To appear. (2012)
4. Ammons, G., Ball, T., Larus, J. R.: Exploiting Hardware Performance Counters With Flow and Context Sensitive Profiling. In Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '97). ACM, Las Vegas, Nevada, USA. 85-96. (1997)
5. Sullins, B. G., Whipple, M. B.: JMX in Action. Manning Publications, USA. (2002)
6. Okanović, D., van Hoorn, A., Konjović, Z., Vidaković, M..: Towards Adaptive Monitoring of Java EE Applications. In Proceedings of the 5th International Conference on Information Technology (ICIT 2011). Al-Zaytoonah University of Jordan, Amman, Jordan. CD. (2011)
7. Okanović, D., Vidaković, M. : Performance Profiling of Java Enterprise Applications. In Proceedings of the International Conference on Internet Society Technology and Management (ICIST 2011). Information Society of Serbia, Kopaonik, Serbia. CD. (2011)
8. Okanović, D., Konjović, Z., Vidaković, M.: Continuous Monitoring System For Software Quality Assurance. In Proceedings of XV International Conference on Industrial Systems (IS'11). University of Novi Sad, Novi Sad, Serbia, 193-198. (2011)
9. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J-M., Irwin, J., Aspect-Oriented Programming. In Proceedings of the European Conference on Object-Oriented Programming. Springer, Jyväskylä, Finland. 220–242. (1997)
10. R Development Core Team. R: A language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. (2010)
11. extremevalues: Univariate Outlier Detection. Mark van der Loo. (2011) [Online] Available: http://cran.r-project.org/web/packages/extremevalues/ (current September 2011)
12. XMLSpy. Altova. [Online] Available: www.altova.com/xmlspy.html (current April 2012)
13. JSR-000914 JavaTM Message Service (JMS) API. Java Community Process. [Online] http://jcp.org/aboutJava/communityprocess/final/jsr914/index.html (current March 2012)
14. The AspectJ Project. Eclipse Foundation. [Online] http://www.eclipse.org/aspectj/ (current April 2012)
15. Lambert, J. M., Power, J. F.: Platform Independent Timing of Java Virtual Machine Bytecode Instructions. Electronic Notes in Theoretical Computer Science, Vol. 220. Elsevier Science Publishers, Amsterdam, Netherlands, 97-113.(2008)
16. Hyperic SIGAR API. Hyperic. [Online] http://www.hyperic.com/products/sigar (current April 2012)
17. Okanović, D., Vidaković, M.: One Implementation of the System for Application Version Tracking and Automatic Updating. In Proceedings of the IASTED International Conference on Software Engineering 2008. ACTA Press, Innsbruck, Austria. 62–67. (2008)
18. EJB 3.0. [Online] Available: http://java.sun.com/products/ejb/ (current April 2012)
19. Barry, D., Stanienda, T.: Solving the Java Object Storage Problem. Computer, Vol. 31, No.11, 33-40. (1998)
20. Gamma, E., Helm, R., Johnson, R., Vlissides, J. M: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Proffesional, Boston, USA. (1994)

21. Kalin, M.: Java Web Services: Up and Running. O'Reilly Media, Sebastopol, California, USA. (2009)
22. Java Swing. Oracle. [Online] Available: http://java.sun.com/javase/6/docs/technotes/guides/swing (current April 2012)
23. Unix Programmer's Manual. Section 1, Bell Laboratories, Murray Hill, NJ. (1979)
24. Srivastava, A., Eustace, A.: ATOM: A System for Building Customized Program Analysis Tools. In Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation. ACM, Orlando, Florida,USA. 196-205. (1994)
25. Snatzke, R. G.: Performance survey 2008. (2008). [Online]. Available: http://www.codecentric.de/export/sites/homepage/__resources/pdf/studien/performance-studie.pdf (current April 2012)
26. Java Virtual Machine Tool Interface (JVMTI). Oracle. [Online] Available: http://download.oracle.com/javase/6/docs/technotes/guides/jvmti/ (current April 2012)
27. Java Virtual Machine Profiler Interface (JVMPI). Oracle. [Online] Available: http://download.oracle.com/javase/1.4.2/docs/guide/jvmpi/jvmpi.html (current April 2012)
28. JBoss Profiler. JBoss Community team. [Online] Available: www.jboss.org/jbossprofiler (current April 2012)
29. Dimitriev, M.: Design of JFluid. Technical Report SERIES13103, Sun Microsystems Inc., USA. (2003)
30. JBoss Application Server. JBoss Community team. [Online] http://www.jboss.org/jbossas (current April 2012)
31. NetBeans. [Online] Available: http://netbeans.org/index.html (current September 2011)
32. Parsons, T., Mos, A., Murphy, J.: Non-Intrusive End-to-End Runtime Path Tracing for J2EE Systems. IEEE Proceedings – Software, Vol. 153, No. 4, 149–161. (2006)
33. dynaTrace – Continuous application performance management. dynaTrace software Inc. [Online] Available: http://www.dynatrace.com/ (current April 2012)
34. JXInsight. JInspired. [Online] Available: http://www.jinspired.com/products/jxinsight/ (current April 2012)
35. IBM - Monitoring Software - Tivoli Monitoring. IBM. [Online] http://www-01.ibm.com/software/tivoli/products/monitor/ (current April 2012)
36. AppDynamics. [Online] Available: http://www.appdynamics.com (current March 2012)
37. Nagios. [Online] Available: http://www.nagios.org (current March 2012)
38. Application Performance Management. CA Technologies. [Online] Available: http://www.ca.com/us/application-performance-management.aspx (current April 2012)
39. HP Systems Insight Manager. Hewlett-Packard. [Online] Available: http://h18013.www1.hp.com/products/servers/management/hpsim/index.html?jumpid=go/hpsim (current April 2012)
40. Newman, H. B., Legrand, I. C., Galvez, P., Voicu, R., Cirstoiu, C.: MonALISA : A Distributed Monitoring Service Architecture. In Proceedings of the Conference for Computing in High-Energy and Nuclear Physics. La Jolla, California, USA. 8pp. (2003)
41. Liu, R., Gibbs, C., Coady, Y.: MADAPT: Managed Aspects for Dynamic Adaptation Based on Profiling Techniques. In Proceedings of the 3rd Workshop on Adaptive and Reflective Middleware. ACM, Toronto, Ontario, Canada. 214 – 219. (2004)

42. Villazón, A., Binder, W., Ansaloni, D., Moret, P.: HotWave: Creating Adaptive Tools With Dynamic Aspect-Oriented Programming in Java. In Proceedings of the 8th International Conference on Generative Programming and Component Engineering (GPCE '09). ACM, Denver, Colorado, USA. 95–98. (2009)
43. Ehlers, J., van Hoorn, A., Waller, J., Hasselbring, W.: Self-Adaptive Software System Monitoring for Performance Anomaly Localization. In Proceedings of the 8th IEEE/ACM International Conference on Autonomic Computing (ICAC 2011). ACM, Karlsruhe, Germany. 197-200. (2011)
44. Object Constraint Language (OCL) 2.0. OMG. [Online] Available: http://www.omg.org/spec/MOF/2.0 (September 2011)
45. Marwede, N., Rohr, M., van Hoorn, A., Hasselbring, W.: Automatic Failure Diagnosis Support in Distributed Large-Scale Software Systems Based on Timing Behavior Anomaly Correlation. In Proceedings of the 2009 European Conference on Software Maintenance and Reengineering (CSMR '09). IEEE Computer Society, Kaiserslautern, Germany. 47-58. (2009)
46. Yu, Y., Rodeheffer, T., Chen, W.: RaceTrack: Efficient Detection of Data Race Conditions via Adaptive Tracking. In Proceedings of the ACM Symposium on Operating Systems Principles. ACM, Brighton, UK. 221-234. (2005)
47. Chen, M., Kiciman, E., Fratkin, E.., Fox, A., Brewer, E.: Pinpoint: Problem Determination in Large Dynamic Systems. In Proceedings of 2002. International Conference on Dependable Systems and Networks. IEEE Computer Society, Washington DC, USA. 595-604. (2002)
48. Aguilera, Mogul, J., Wiener, J., Reynolds, P., Muthitacharoen, A.: Performance Debugging for Distributed Systems of Black Boxes. In Proceedings of the 19th ACM symposium on Operating systems principles. ACM, Bolton Landing, New York, USA. 74-89. 2003.
49. Wang, H., Platt, J., Chen, Y., Zhang, R., Wang, Y.: PeerPressure for Automatic Troubleshooting. In Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems. ACM, New York, New York, USA. 398-399. (2004)
50. SPECjvm98. Standard Performance Evaluation Corporation. (1998) [Online] Available: http://www.spec.org/jvm98/ (current 12 September 2011)
51. Govindraj, K., Narayanan, S., Thomas, B., Nair, P., Peeru, S.: On using AOP for Application Performance Management. In Industry Track Proceedings of the 5th International Conference on Aspect-Oriented Software Development. ACM, Bonn, Germany. (2006)
52. Tebbani, B., Aib, I.: GXLA a Language for the Specification of Service Level Agreements. Lecture Notes in Computer Science, Vol. 4195. Springer-Verlag, Berlin Heidelberg New York, 201-214. (2006)
53. Keller, A., Ludwig, H.:The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. Journal of Network and Systems Management, Vol. 11, No. 1, 57-81. (2003)
54. Lamanna, D., Skene, J., Emmerich, W.: SLAng: A Language for Defining Service Level Agreements. In Proceedings of the 9th IEEE Workshop on Future Trends of Distributed Computer Systems (FTDCS '03). IEEE Computer Society, San Juan, Puerto Rico. 100-107. (2003)
55. Meta Object Facility (MOF) 2.0 Core Specification. OMG. [Online] Available: http://www.omg.org/spec/MOF/2.0 (current September 2011)
56. Human Usable Textual Notation (HUTN) Specification. OMG. [Online] Available: http://www.omg.org/spec/HUTN/index.htm (current September 2011)

57. Oldham, N., Verma, K., Sheth, A., Hakimpour, F.: Semantic WS-agreement partner selection. In Proceedings of the 15th International Conference on World Wide Web. ACM, Edinburgh, Scotland, UK. 697-706. (2006)
58. Paschke, A., Schnappinger-Gerull, E.: A Categorization Scheme for SLA Metrics. In Proceedings of Multi-Conference Information Systems. Passau, Germany. (2006)

Dušan Okanović is a teaching assistant and PhD student atthe Faculty of Technical Sciences, Novi Sad, Serbia. He received his Bachelor degree (2002) and Masters degree (2006), both in Computer Science from the University of Novi Sad, Faculty of Technical Sciences. His research interests include application management, performance management and distributed applications development. Since 2003 he has been with Faculty of Technical Sciences where he was teaching where he participated in several science projects and published 25 scientific papers. His research interests are web and internet programming, distributed applications, application management, and performance management. He can be contacted at: oki@uns.ac.rs.

André van Hoorn is a research assistant and PhD student with the Software Engineering Group at the University of Kiel, Germany. He received his Diploma (Master equivalent) degree in Computer Science from the University of Oldenburg, Germany (2007). From 2008 to 2010, André was member of the Graduate School on Trustworthy Software Systems (TrustSoft) at the University of Oldenburg, where he was holding a PhD scholarship from the German Research Foundation (DFG). Since 2011, he works in the collaborative research project DynaMod on dynamic analysis for model-driven software modernization. His research interests include architecture-based and model-driven software performance engineering, self-adaptation, and reengineering. He published more than 20 scientific papers. André can be contacted at: avh@informatik.uni-kiel.de.

Zora Konjović has been holding the full professor position at the Faculty of Technical Sciences, Novi Sad, Serbia since 2003. Mrs. Konjović received her Bachelor degree in Mathematics from the University of Novi Sad, Faculty Science in 1973, Master degree in Robotics from the University of Novi Sad, Faculty of Technical Sciences in 1985, and Ph. D. degree in Robotics from the University of Novi Sad, Faculty of Technical Sciences in 1992. From 1973 till 1980 she was with the Faculty of Science in Novi Sad, and since 1980 she has been with the Faculty of Technical Sciences, University of Novi Sad. Mrs. Konjović participated in 5 scientific and more than 30 professional projects; in 5 she was the project leader. She published more than 150 scientific and professional papers. She is the corresponding author and can be contacted at: ftn_zora@uns.ac.rs.

Milan Vidakovic received the BSc, MSc and PhD degrees in electrical engineering from the Faculty of Technical Sciences, University of Novi Sad, in 1995, 1998 and 2003 respectively. He is a professor at Computing and Control Department, University of Novi Sad. He participated in several science projects and published more than 60 scientific and professional papers. His research interest covers web and internet programming, distributed computing, software agents, embedded systems, and language internationalization and localization. He can be contacted at: minja@uns.ac.rs.

Received: September 26, 2011; Accepted: June 14, 2012

DOI: 10.2298/CSIS111029039A

A Scalable Multiagent Platform for Large Systems

Juan M. Alberola, Jose M. Such, Vicent Botti,
Agustín Espinosa and Ana García-Fornes

Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València Camí de Vera s/n. 46022, València (Spain)
{jalberola,jsuch,vbotti,aespinos,agarcia}@dsic.upv.es

Abstract. A new generation of open and dynamic systems requires execution frameworks that are capable of being efficient and scalable when large populations of agents are launched. These frameworks must provide efficient support for systems of this kind, by means of an efficient messaging service, agent group management, security issues, etc. To cope with these requirements, in this paper, we present a novel Multiagent Platform that has been developed at the Operating System level. This feature provides high efficiency rates and scalability compared to other high-performance middleware-based Multiagent Platforms.

Keywords: Multiagent Platforms, Multiagent Systems, Evaluation.

1. Introduction

In the last decade, due to the rapid growth of the Internet, the speed of change, and an ever greater amount of easily accessible information, the next generation of Multiagent Systems (MAS)s and information technology, will target open and large systems. In these dynamic and heterogeneous environments, it is essential that features such as security, high performance, scalability, and interoperability are provided by application development frameworks.

Even though current Multiagent Platforms (MAP)s support the development and execution of MASs, very few real applications have been developed to focus on open and dynamic systems. These applications change quickly and require features such as reliability, scalability, and performance, which not many MAPs are designed to offer. According to [25], agent researchers should design and implement large software systems consisting of hundreds of agents and not only systems composed of a few agents. In order to develop these systems, researchers require efficient and scalable MAPs.

Some current MAPs are not suitable for executing complex systems because their designs are not oriented to improving efficiency and scalability issues. Previous studies have demonstrated a degradation in the performance of current MAPs as the system grows [51, 22]; some MAPs even fail [49]. Our main objective for this paper is to propose a MAP that is focused on being scalable and efficient. One of our main design decisions is to use the operating

system (OS) services to develop this MAP instead of using middlewares between the OS and the MAP. In [14] we proved that this can noticeably improve the performance and scalability of the system.

Functionality is another important issue when executing large systems. Works by other researchers such as [20] are helpful in determining the main requirements for designing a MAP. By using theoretical proposals and methodologies [27], a MAP that supports agent organizations helps to simplify, structure, coordinate, and easily develop large applications, which are composed of thousands of agents. Standard language communication is another key requirement for allowing the interaction between heterogeneous entities. Support to coordinate communication is another requirement for these systems [42]. Definition of standard speech acts that agents can use, a common ontology to describe and access services, policies associated to agent conversations, and standard communication language are some features that should be provided. Finally, security concerns become important in large systems must be addressed if these systems are open in order to ensure the communications and the identities of each entity. As stated by other authors in [45], these features should be provided by agent execution frameworks.

Towards these goals, in this paper, we present a MAP that is oriented to fulfilling the requirements for this new kind of systems. This MAP is mainly focused on scalability and efficiency for executing large MASs. It provides mechanisms to support agent organizations, security concerns (authentication, authorization, and integrity), a standard language of communication for information representation, conversation-oriented interactions, and so on.

The rest of the article is organized as follows. Section 2 presents the motivation and the previous work that allowed us to design and develop an efficient and scalable MAP. Section 3 gives an in-depth description of the MAP architecture. Section 4 details the services offered by the MAP. Section 5 describes how agents in this MAP are represented. Section 6 describes a tourism service application that is built on this MAP. Section 7 presents a performance evaluation of the MAP. And finally, in Section 8, we present some concluding remarks.

2. Motivation and previous work

In the last few years, many researchers have focused on testing the performance of existing MAPs. One of the main properties tested in these works is the performance of the MAPs for sending messages. Vrba [51] presents an evaluation of the messaging service performance of four MAPs. From the tests presented in that paper, the author concludes that Jade [19] provides the most efficient messaging service compared to FIPA-OS [1], Jack [3], and ZEUS [12]. However, the design features that produce this performance are not given and the implementations of the messaging service for each MAP are not detailed. Therefore, these conclusions can only be valid to choose the MAP that performs better than the other three MAPs tested. Burbeck et al. [22] tested the messaging service performance of three MAPs. They claim that Jade performs better

than Tryllian [11] and SAP [9] because it is built on Java RMI[1], but they give no proofs confirm this claim. As these works state, Jade is more scalable than other MAPs and can be considered to be a stable MAP for large systems [40]. However, these conclusions do not provide any clue to MAP developers about how to improve MAP designs since these experiments only scale up to 100 pairs of agents and a few hosts. A more thorough study is required to be able to assess MAP performance and to determine to what extent design decisions influence MAP performance.

Some other works have tested the performance of other services but only for a single MAP. Most of these works test Jade, which seems to be the most widely used MAP. In [25], the authors tested Jade messaging, agent creation, and migration services. The tests that they performed related to the messaging service only scale up to eight agent pairs. In [17], an evaluation of a MAS for adapting application's behaviour was carried out on Jade MAP. The work presented in [26] tested the scalability and performance of the Jade messaging service. Similar to the works cited above, their conclusions do not provide any design decision. Even though these conclusions can allow MAS developers to check whether or not Jade fulfills their requirements when designing a MAS, they do not suggest any design decision for MAP developers.

There are also other works that focus on testing the performance of a specific MAS that is running on top of a MAP. In [23] the performance of MAPs is measured when a MAS composed of several web agents is launched. This MAS provides documents requested by a user agent. The authors measured the number of documents requested per unit of time. Therefore, their conclusions are only valid for this MAS. Lee et al. [37] present a MAS in which agents coordinate with each other to carry out tasks. They evaluate how the topological relations between agents affect the number of CPU cycles needed to accomplish these tasks. In [28], the authors compare the response time and the CPU cycles of SACI [13] and Jade.

Finally, other studies focus on detailing the functional properties of MAP. In [20], four MAPs are compared according to several criteria: implementation languages, tools provided, agent deliberation capabilities, etc. Shakshuki [46] presents a methodology to evaluate MAPs based on several criteria such as availability, environment, development, etc. Similar work is carried out by Nguyen [33], and Omicini [43] gives a brief evolution of MAPs. In other works such as [34, 44], different MAPs that are intended to be scalable are proposed; however, no empirical evaluation is carried out. These works provide ratings of properties provided by MAPs in order to help users choose the MAP according to their needs. Our work goes a step further since it is not only intended to be useful for MAP users but also for MAP developers.

A general conclusion of works that focus on MAP evaluation is that MAP performance decreases as the system grows. Furthermore, as we showed in a previous work [49], when large-scale MAS are taken into account, the performance of many MAPs is considerably degraded when the size of the system

[1] http://java.sun.com/docs/books/tutorial/rmi/index.html

executed increases, causing some MAPs to even fail. Therefore, current MAPs are not suitable for executing large population systems because their designs are not aimed at improving efficiency and scalability issues.

In order to develop a design in accordance with our goals, we detail other previous works that we carried out that focus on finding design decisions that influence MAP performance. In [41], we presented experiments to link performance with internal MAP designs, that is, to identify the key design decisions that lead to better performance. We extracted some conclusions from these experiments, such as the fact that centralizing services in a single host of the MAP degrades the performance causing this host to become a bottleneck in the case of very popular services. It is more suitable to design a distributed approach with efficient information replication mechanisms. In [16], we tested several issues of the MAPs, such as the performance of the directory service proposed by FIPA [2], the memory consumed by the agents and the MAP, the network occupancy rate, the CPU cycles, etc. According to these studies, the most influential point in the MAP performance that could become a bottleneck is the messaging service. This service is crucial in the performance of the MAP since agents need to exchange messages with other agents and access MAP services. Furthermore, some MAPs (such as Jade) base other MAP services (such as the Agent Directory or Service Directory proposed by FIPA) on the messaging service. In [14], we specifically analyzed technologies for implementing the Message Transport System (MTS), which is the component of the MAP that manages the message exchanges among the agents running on the MAP. This work showed that in order to design a messaging service that can handle large agent populations, the design that performs better should be based on direct communication between each pair of agents so that the messaging service scales better and performs more efficiently, especially in these sorts of scenarios.

In the following sections, we present a MAP focused in being scalable and efficient in more detail. It has been developed using the services offered by the OS to support MAS efficiently. By bringing MAP design closer to the OS level we can define a long-term objective, i.e., to incorporate the agent concept into the OS itself in order to offer a greater abstraction level than current approaches.

3. Magentix Multiagent Platform architecture

Magentix[2] MAP aims to be scalable and efficient, mainly when it is executing large-scale MAS. To achieve a response time closer to the achievable time lower bound, this MAP has been developed using the services provided by the OS. Thus, one of the design decisions is that this MAP is written in C over the Linux OS. Current approaches for developing MAPs are based on interpreted languages like Java or Python. These MAP designs are built over middlewares like the Java Virtual Machine (JVM) [21]. Although these middlewares offer some advantages like portability and easy development, MAPs developed over them

[2] Magentix can be downloaded from http://gti-ia.dsic.upv.es/sma/tools/Magentix/index.php

do not perform as well as one might expect, especially when they are running large systems. In [14] we presented a performance evaluation related to this issue. We proved in that using the Operating System (OS) services to develop a MAP instead of using middlewares between the OS and the MAP noticeably improves the performance and scalability of the MAP. Thus, we can see the MAP functionality as an extension of the functionality offered by the OS.

The Magentix communication service has been developed to offer high performance. This service is quite crucial to the performance of the MAP as we stated in Section 2 and some other services may be implemented using it. Magentix also provides advanced communication mechanisms such as agent groups, a manager to execute interaction protocols, and a security mechanism to provide authentication, integrity, confidentiality, and access control. This design has been developed in order to provide the functionality required by MAS and perform efficiently.

Magentix is a distributed MAP composed of a set of computers executing Linux OS (figure 1). Magentix uses replicated information on each MAP host to achieve better efficiency. Each one of these computers presents a process tree structure. The initial design of this structure is presented in [15]. The advantage of process tree management offered by Linux, and using some services like signals, shared memory, execution threads, sockets, etc. provides a suitable scenario for developing a robust, efficient, and scalable MAP.

The structure of each Magentix host is a three-level process tree. On the higher level we see the *main* process. This process is the first one launched on any host when this host is added to the MAP. Below this level we can see the services level. Magentix provides some services to support agent execution: Agent Management System (*AMS*), Directory Facilitator (*DF*), and Organizational Unit Manager (*OUM*). Services are represented by means of service agents replicated in every MAP host. Agents representing the same service manage replicated information and communicate with each other in order to keep this information updated. Finally, in the third level, user agents are placed. Using this process tree structure, *main* process manages service agents completely, i.e., it can kill any service agent to achieve a controlled shutdown of the MAP, and also detects at once whether any service agent dies. In the same way, *ams* agent has a broad control of the user agents of its own host.

Each user agent is represented by a different Linux child process of the *ams* agent running on the same host. This design decision was taken after efficiency tests as we stated in Section 2. Mapping one-to-one agents and Linux processes provides us with a complete execution control (as we will see in the next section) and a fast message exchanging mechanism. It could be argued that using a single virtual machine for executing agents represented as Java threads could be lighter. Nevertheless, this virtual machine could be overloaded when running three or four thousand agents, by the limitations of the virtual machine. In our proposal, mapping agents as Linux processes restricts us to the limitations of the OS, and allows us to run more than seven thousand agents in a single host. Developing a MAP by using the OS services directly allow us to

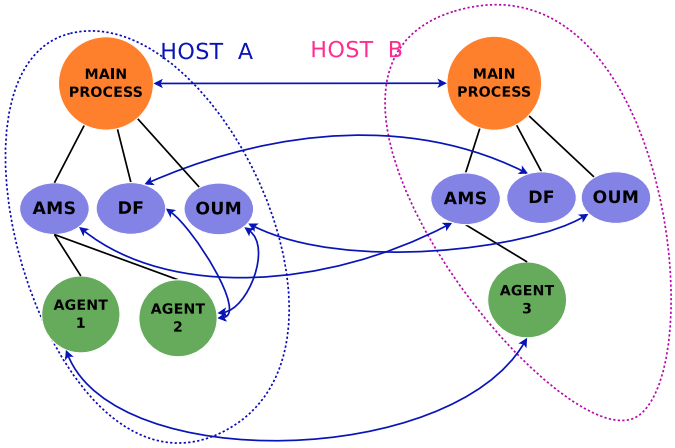


Fig. 1. Platform structure: Agent Management System (*AMS*), Directory Facilitator (*DF*), Organizational Unit Manager (OUM)

improve the efficiency of the system. Current Magentix version offer support to different Linux distributions (such as Ubuntu, Fedora, CentOs or OpenSuse) as well as to Mac OS. Interoperability between heterogeneous agents is reached by means of standard communication language representation and ontologies for service interactions.

3.1. Communication and Message Transport System

Magentix provides a message-based communication mechanism in order to allow interactions between agents and services. This communication mechanism aims to obtain both good efficiency level and MAP scalability. As Magentix MAP is integrated into Linux, we have checked different alternatives available for communicating processes in an OS context [14]. In this study we have analyzed different communication services among processes provided by POSIX [10] compliant OS, in particular, the Linux OS, in order to select which of these services allows robust, efficient and scalable MAPs to be built. As a result of the evaluation, a lower bound of the time needed to communicate process couples (located in the same or different hosts) was obtained. In these studies, we showed to what extent the performance of a Message Transport System (MTS) degrades when its services are based on middlewares between the OS and the MAP (like the JVM) rather than directly by the underlying OS. Thus, the Magentix MTS design was tested to be as close as possible to this time lower bound.

As we pointed out in section 2, the messaging service design that should perform better would be one based on direct communication between each pair of agents. Therefore, the communication mechanism implemented in message exchanging interactions is carried out by means of point to point connec-

tions based on TCP sockets, between a pair of processes. This mechanism enables high scalability in agent communication. Each Magentix agent has a server socket for receiving connections from other agents by means of client sockets. To carry out a new connection an agent creates a client socket that communicates with the remote agent server socket. Thus, Magentix agents are client/server at the same time.

At a lower level, Java-RMI technology (used for development communication in most of MAPs based on Java) uses TCP sockets. After evaluating different alternatives, we finally define the communication mechanism implemented in Magentix as point to point connections based on TCP sockets, between a pair of processes. The use of C language to develop the MAP, allows us to use this technology closer to the OS level, and avoid the overhead resulting from the use of Java-RMI, because the agent abstraction provided by a MAP is independent of the underlying communication mechanism implementation.

In our previous studies, we have also checked that opening a P2P connection between a pair of agents the first time they interact and leaving this connection open for future interactions is much more efficient than opening a new TCP connection each time they want to interact. Therefore two agents could have an indefinitely open connection for exchanging messages each time they require it. Nevertheless, the number of simultaneous open connections is limited by the OS. Therefore, each agent and service stores its open connections in a *connection table*. The first time an agent contacts another one, a TCP connection is established and remains open to exchange messages in the future. These connections are automatically closed when the conversation is not active, that is, some time has passed since the last message was sent, according to a LRU (Last Recently Used) policy (this mechanism is described in more depth in [50]). This *connection table* improves communication times since an agent does not need to create a new TCP connection each time it wants to communicate with another agent.

4. Services

In this section we describe the services that are implemented in Magentix oriented to agents, services, and group management: *AMS* service, *DF* service and *OUM* service.

4.1. Agent Management System

Agent Management System (*AMS*) service is defined by FIPA [29] and offers the white pages functionality. This service stores the information regarding the agents that are running on the MAP. *AMS* service is distributed among every MAP host. Therefore, information regarding the agents of the MAP is replicated in each host. This service is represented as *ams* agents running in each host of the MAP.

As we stated in section 3, all of the agents launched in a specific host are represented by means of child processes of the *ams* agent. Just as the *main* process behaves, the *ams* agent has a broad control of the agents in its corresponding host. The management of starting and finalizing agents is automatically carried out by means of sending signals

The *AMS* service stores the information regarding every agent running on the MAP. This service allows us to obtain the physical address (IP address and port), providing the agent name to communicate with. Due to the fact that the *AMS* service is distributed among every MAP host, each *ams* agent running on each host contains the information needed to contact every agent of the MAP. Hence, the operation of searching agent addresses is not a bottleneck as each agent looks this information up in its own host, without needing to make any requests to centralizing components. Every time an agent is started or finalized in a host, this update is replicated on each host of the MAP. Nevertheless, there is another information regarding agents that does not need to be replicated when it is updated. For this reason, the *ams* agents manage two tables of information: the Global Agent Table (*GAT*) and the Local Agent Table (*LAT*).

- **GAT**: Stored in this table is the name of each agent in the MAP and its physical address, that is, its IP address and its associated port.
- **LAT**: In this table additional information is stored such as the agent's owner, the process PID which represents each agent and its life cycle state.

The *GAT* is mapped on shared memory. Every agent has read only access to the information stored in the *GAT* of its own host. Each time an agent needs to obtain the address of another agent in order to communicate, it accesses the *GAT* without making any request to the *ams* agent. Thus, we avoid the bottleneck of requesting centralizing components each time one agent wants to communicate with another. The information contained in the *GAT* needs to be replicated in each host to achieve better performance. Although replication mechanisms imply an overhead in the system, this overhead is reduced as only the updated information is replicated, and these updates occur when agents are started or dead in the MAP, operations that generally occur in low frequency rates. Thus, the overhead resulting from replication is worthwhile in order to distribute the information and make it available in each host of the MAP. Moreover, the spacial overhead (memory) for having the same information replicated in each host is also low, due to the fact that only the physical addresses of the agents are distributed (few bytes of memory).

The information from the *LAT* is not replicated. Some information stored in the *LAT* regarding a specific agent is only needed by the *ams* agent of the same host (for instance, the process PID). Therefore, this information does not need to be replicated. Some other information could be useful for the agents but is not usually requested (such as the life cycle state). In order to reduce the overhead resulting from replication, we divide the information regarding agents into two tables. Each *ams* stores in their *LAT* the information regarding the agents under its management, that is, the agents that are running on the same host. If some information available to agents is needed (such as the life cycle state), the agent

has to make a request to the *AMS* service using the *AMS* service ontology. In a transparent way, these requests addressed to the *AMS* service are delivered to the specific *ams* running on the same host as the agent requested.

4.2. Directory Facilitator

The Directory Facilitator (*DF*) service offers the yellow pages functionality defined by FIPA. This service stores the information regarding the services offered by agents. The *DF* service allows agents to register the services they provide, deregister these services, and look up a specifically required service. Much like the *AMS* service, the *DF* service is implemented in a distributed scenario by means of agents running on each MAP host, called *df* agents. Information regarding services is also replicated in every host of the MAP.

Information that needs to be replicated is stored in a unique table called **GST** (Global Service Table). This table is a list of pairs: services offered by agents of the MAP and the agent that offers this service. In contrast to the **GAT**, the **GST** is not implemented as shared memory, therefore only the *df* can acces this information directly.

Agents are able to register, deregister, and look up services offered by other agents. To do these tasks, agents need to communicate with the *DF* service using the *DF* service ontology. Current functionality of the *DF* service is the one proposed by FIPA. Nevertheless, we consider the possibility of improving this service in order to provide new functionalities such as the management of semantic information, service composition, services offered by agent organizations, etc. and also extending the operations proposed by FIPA for registering, deregistering, and searching for services.

4.3. Organizational Unit Manager

The Organizational Units Manager (*OUM*) service provides support oriented to agent-group communication as a pre-support for agent organizations. Several research groups define theoretical proposals and methodologies to design MASs, oriented to organizational aspects of the agent society [27]. In order to develop applications which use these organization oriented methodologies, we require MAPs that support them. Among there are few MAPs which offer any kind of support related to agent organizations. Among these MAPs are Jack [3], MadKit [4], or Zeus.

An agent group in Magentix is called organizational *unit* (from now on, unit) and can be seen as a blackbox from the point of view of external agents. Units can also be composed of nested units. Agents can interact with an agent unit in a transparent way, i. e. from the point of view of an agent outside the unit, there is no difference between interacting with a unit or with an individual agent. Interaction between an agent and a unit is carried out by the MAP through properties specified by the user. Each unit has some properties associated to it. As each agent of the MAP has a unique name, each unit is identified in the MAP by its *name*. In order to interact with any unit, user must specify one or

more agents to receive the messages addressed to the unit: these agents are called *contact agents*. User can also specify the way in which these messages have to be delivered to the *contact agents*. This property is called the *routing type* and messages addressed to the unit will be delivered to the contact agents defined according to one of these *routing types*:

- Unicast: The messages addressed to the unit are delivered to a single agent which is responsible for receiving messages. This type is useful when we want a single message entrance to the group. It could be useful if the group has for example, a hierarchical structure, where the supervisor receives every message and distributes them to its subordinates.
- Multicast: Several agents can be appointed to receive messages. When a message is addressed to the unit, this message is delivered to any contact agent in the unit. This could be useful if we want to represent an anarchic scenario, where every message needs to be known by every agent without any kind of filter.
- Round Robin: There can be several agents appointed to receive messages. But each message addressed to the unit is delivered to a different contact agent, defined according to a circular policy. This type of routing messages is useful when several agents offer the same service but we want to distribute the incoming requests to avoid the bottlenecks.
- Random: Several agents can be defined as contact agents. But the message is delivered to a single one, according to a random policy. As with the previous type, this is useful for distributing the incoming requests, but no kind of order for attending these requests is specified.
- Sourcehash: Several agents can be the contact agents. But any given message is delivered to one of the agents responsible for receiving messages, according to the host where the message sender is situated. This is a load-balancing technique.

Units have a defined set of agents which make up the unit, called *members*. These agents can interact and coordinate with each other and each one plays a certain *role*. Finally, each unit has a *manager* associated to it. This agent is responsible for adding, deleting or modifying the members and contact agents. By default it is the agent which creates the unit and is the only one allowed to delete it.

All of this information regarding units in Magentix, is managed by the *OUM* service, which stores it in the **GUT** (Global Unit Table). Similar to the previous services, *OUM* is a distributed service composed by *oum* agents running on each MAP host. The **GUT** table is replicated and synchronized on each host of the MAP every time an update is made. Interaction between agents and *OUM* service is carried out by the sending of messages using the *OUM* service ontology.

4.4. RDF as framework for representing information

To develop large systems, standard language communication is a key requirement for allowing the interaction between heterogeneous entities. FIPA pro-

poses some Agent Communication specifications regarding the language used for message exchanging in a MAP [30]. They standarize the structure of an Agent Communication Language (ACL) message to ensure interoperability and also Content Language (CL) specifications for representing the content of the ACL messages. The use of standard specifications is vital in order to allow interoperability between heterogeneous agents which could compose an open system, as well as to define standard ontologies for accessing the MAP services.

Resource Description Framework (RDF) is a language for representing information about resources on the World Wide Web. By generalizing the concept of a "Web resource", RDF can also be used to represent information about things even when they cannot be directly retrieved from the Web [6]. RDF is based on the idea of identifying things using Web identifiers (called Uniform Resource Identifiers, or URIs), and describing resources in terms of simple properties and property values. The underlying structure of any expression in RDF is a collection of triples, each consisting of a subject, a predicate and an object. The subject can be any resource, the predicate is a named property of the subject and the object denotes the value of this property. A set of such triples is called an RDF graph. RDF also provides an XML-based syntax (called RDF/XML [7]) for recording and exchanging these graphs. RDF is intended for situations in which this information needs to be processed by applications, rather than only being displayed to people. RDF provides a common framework for expressing this information so it can be exchanged between applications without loss of meaning.

Due to the features of RDF and its widespread use in MAS [18, 24, 35, 36], an RDF-based framework for managing information has been designed for Magentix and has been integrated into it. It allows a Magentix agent to manage all of its information as RDF models (RDF graphs). Moreover, Magentix itself uses the framework for the messages exchanged, for representing the information that Magentix services manage, for interacting with the Magentix services and for storing MAP events.

The framework is based on offering an API to deal with RDF management. Of course, we did not implement an RDF support from scratch, the framework is designed as a wrapper for existing RDF management libraries and is aimed at simplifying the use of RDF inside a Magentix agent. There are some libraries that deal with RDF models. However, because of the Magentix features, i. e., the fact that it is implemented in C language and is focused on achieving high levels of efficiency, we have chosen the Redland libraries [8].

Redland is a set of free software C libraries that provide support for RDF. The authors of Redland claim that it is portable, fast and with no known memory leaks. It allows the manipulation of the RDF graph, triples, URIs and Literals. It can be implemented efficiently in C, providing memory storage with many databases (Berkeley DB, MySQL, etc.). We use the RDF/XML syntax to serialize the RDF graphs, but Redland also support other syntaxes, such as N-Triples

or Turtle Terse RDF Triple Language. Queries can be carried out with SPARQL or RDQL.

One of the functionalities of Magentix where the RDF has been used in it is to represent messages. Agents and services use message sending to communicate with each other as we said in section 3.1. FIPA defines the structure of an Agent Communication Language (ACL) and also defines the use of RDF to represent the message content [31]. Message header and message content in Magentix are represented as RDF models serialized as XML. Some MAPs use this kind of serialization to represent the message content only (such as Jade), just as FIPA proposes. We provide Magentix with RDF to represent the whole message. Therefore, only one parser is needed and this simplifies the parsing and serializing process of a message.

As far as we are concerned, representing the FIPA-ACL using RDF should be standard, but currently it is not, so interoperability with other FIPA-compliant MAPs is compromised. A simple gateway that directly translates both representations can be added to solve this problem. Figure 2 shows an example of a Magentix message. It is an RDF graph in which resources are drawn as ellipses and literals are drawn as squares. As can be observed, all of the FIPA-ACL fields are mapped as RDF properties describing a message resource. The content of the message can also be seen as an RDF sub-graph inside the main RDF graph representing the message. Therefore, any information that a Magentix agent has as an RDF graph, can be added or retrieved directly from a message.

Regarding the representation of information about Magentix services, an ontology for interacting with them has been defined using Web Ontology Language (OWL) [5]. The ontology mainly focuses on describing the resources that the services manage (hosts, agents, services, organizational units, etc.). Therefore, all of the information is treated, without taking into account implementation concerns, so that a change in the implementation does not have any effect on the way the services treat the information. What is more, they can store all of its information in a direct and simper fashion on a database.

In order to achieve rich and flexible interactions between agents and Magentix services, the ontology also includes actions that can be requested by a Magentix service (creation of a new agent to the *AMS*, registering a service to the *DF*, creation of a new organizational unit to the *OUM*, etc.). Therefore, any Magentix agent that knows the ontology can interact with Magentix services and also manage all of the related knowledge using the framework provided.

4.5. Security Model

The Magentix MAP has a security model [48, 47], which is based on both the Kerberos protocol and the Linux OS access control. This model provides Magentix with authentication, integrity and confidentiality. By means of this model each agent has an identity which it can prove to the rest of the agents and services in a running Magentix MAP.

Magentix agents can have three identity types:

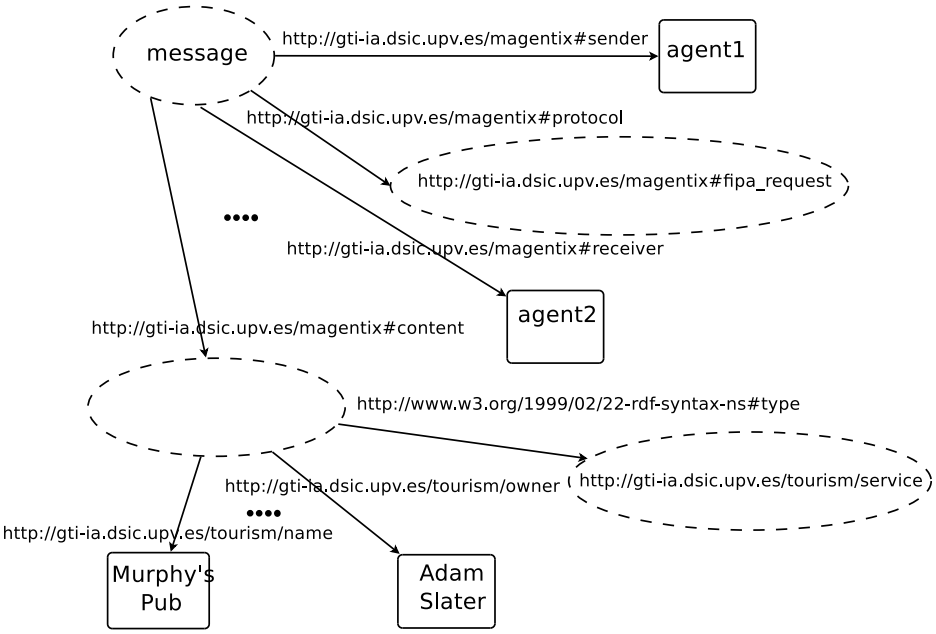


Fig. 2. Magentix Message represented in RDF

- **Agent** identity. Its identity as an agent. This identity is created by the AMS when the agent is created.
- **User** identity. The identity of its owner, i.e, the identity of the user that created the agent.
- **Unit** identity. The identity of each unit that the agent is in.

An agent always has at least its **Agent** identity and its owner's **User** identity. Therefore, a Magentix agent is provided with more than one identity, so a way of letting the Magentix communication module know which Kerberos credentials it has to use when sending a message is needed. This is done with a new field in the message header. If this field is in the message header of a message to be sent, the communication module tries to use the identity chosen; otherwise the corresponding agent identity is used. If the Kerberos credentials associated to the identity that the agent is requesting are not available, and the agent is trying to use an identity that it does not own for instance, the sending of the message fails.

Magentix services are based on information replication in each host. In order to check the integrity of this information and protect it from being accessible to non-authorized users, service communication needs to be secured. In order to do so, the administrator creates a *principal* (the *principal* is the unique name of a user or service allowed to authenticate using Kerberos) for each service with a random key that is saved by default in /etc/krb5.keytab. This file is secured

using Linux OS access control and can only be accessed by the *root* user, so Magentix services have to run as *root* privileges.

When a service requires communication with another service, a security context is established as a client with the *principal* of the MAP administrator and as a server with the *principal* of the destination service. Using this security context the information sent is encrypted and a message integrity code is calculated. Therefore, the client is sure that the destination service is the service expected. Moreover, the destination service knows that it is being contacted by a service with the administrator's identity, so the destination service will serve all of the requests it receives. Thus, only MAP services can exchange information with each other.

Securing agent communication is similar to securing service communication, but agents use the identity that the `ams` agent has created for them when creating a security context to allow a secure interaction with each other.

In order to make efficient use of security contexts, a context cache has been added to each agent. This cache contains the corresponding security context associated with a destination agent. This cache is not related to the connections cache, so that, when a connection with an agent is closed, the associated security context is not lost.

5. User Agents

Agents in Magentix are represented as Linux processes. Internally, every agent is composed of Linux threads: a single thread for executing the agent tasks (main thread), a thread for sending messages (sender thread) and a thread for receiving messages (receiver thread). The *ams* agent manages the creation and deletion of the user agents launched on the same host. The GAT is shared between the *ams* agent and these user agents, so accessing the physical addresses of any agent of the MAP is fast and does not become a bottleneck. Agents have free read access to the GAT, thus, searching for the address of any agent registered in the MAP is efficient.

Magentix provides a template for developing agents written in C++. We provide different methods to manage the agent execution life cycle as well as the message sending and reception. Furthermore, agent developers can extend this model to include other requirements. Interaction with services is easily carried out by means of a specific API. Interactions among agents are focused on conversations. An agent can be interacting with several agents or services at any time. Each interaction between two agents can be represented as a pattern of communication where some messages are exchanged between the participants. These patterns can be predefined or not, but there is an initial message and a final message. The entire amount of messages exchanged between two participants represents a conversation. Magentix provides two functionalities for managing conversations: mailboxes and conversation managers.

5.1. Mailboxes

Mailboxes are used to improve the management of incoming messages from any agent. An agent is able to interact simultaneously with several agents. In these scenarios the possibility of distributing the incoming messages in different message queues, depending on the conversation that belongs each message, becomes interesting. By default every agent has a unique Mailbox called *DEFAULT_MAILBOX*, which receives every message addressed to the agent.

Magentix allows agent developer to create new Mailboxes and later, associate a conversation identifier to them. Then, when a message with this conversation identifier (represented as the *conversation_id* field of the message) is received, this message is routed to the corresponding Mailbox. This functionality allows messages to be filtered and split according to this field, so that, agent developers can easily distribute the different conversations which an agent is involved in among different Mailboxes. A Mailbox is not restricted to receive only the messages of a specific conversation identifier since we consider the possibility of associating several identifiers to the same Mailbox. The basic functionality an agent developer needs to bear in mind when working with Mailboxes is creating new Mailboxes and then, associating them to conversation identifiers. When an agent checks the message incoming queue, it specifies which Mailbox it wants to check. We can see in figure 3 an image of the internal structure of a Magentix agent.

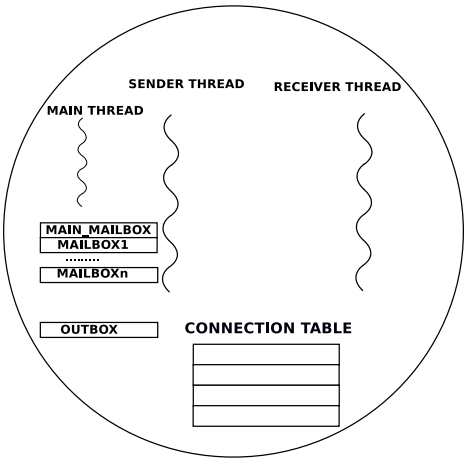


Fig. 3. Magentix Agent

5.2. Conversation Manager

Interactions between Magentix agents are focused on conversations. Thus, it is important for us not only to the searching and sending of messages to other agents but also to easily reproduce typical conversation patterns that can appear in a big variety of scenarios. An agent is able to simultaneously communicate with several agents. Every interaction between a pair of agents very often requires the exchanging of more than one message. Moreover, message exchange patterns are usually repeated in several interactions between agents, i.e. to access some service, to request information, to send proposals to different agents, etc. Thus, defining communication patterns to specify which messages exchanges are allowed for a specific interaction proves to be an interesting and useful feature for agent developers.

FIPA defines standard interaction protocol specifications that agents can use in their conversation with other agents ([32]). These specifications deal with pre-agreed message exchange protocols for ACL messages. Magentix provides support for executing these protocols defined by FIPA, therefore, agent developers can easily reproduce these interaction scenarios without needing to consider the sequence of exchanged messages, the possible failures in the execution of the protocol and so on. Agent developer only has to specify what to do when some of the deterministic events of the protocol take place and the protocol will automatically be checked and executed by Magentix.

Interaction protocols are defined by FIPA using UML-diagrams. In figure 4 we can see the protocol FIPA-request as an example. In these protocols there are two roles, initiator and participant, which exchange some possible message sequences. We translate this representation to Magentix as finite state machines. Each interaction protocol has a finite state machine associated to each possible role of the protocol. In figure 5 we can see the FIPA-request protocol for the initiator role. Each finite state machine has these properties:

- A *not create* initial state. This state is the first of every protocol.
- Transitions which allow the execution of the protocol depending on the messages received (represented as performatives such as refuse or agree) or λ-transitions, which take the protocol execution forward to the next state.
- Intermediate states for representing the intermediate steps of the protocol execution.
- A *delete* state. This is the last one of every protocol.

In order to process these interaction protocols we define a conversation manager. A conversation manager is an internal entity within Magentix agents, which has one or more interaction protocols associated to it. When an agent is using one of these protocols in its conversations with other agents, its conversation manager is in charge of automatically managing it and ensuring the correct execution of the protocol, executing each step and transition of the protocol. Several conversation managers can be assigned to a single agent, each one in charge of the management of different interaction protocols. This decision depends on the agent developer, which can run more conversation managers or

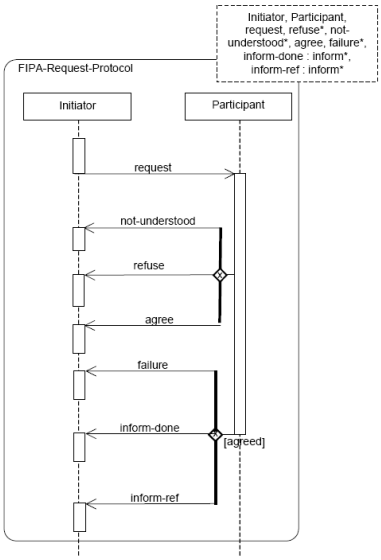


Fig. 4. FIPA-request Interaction Protocol

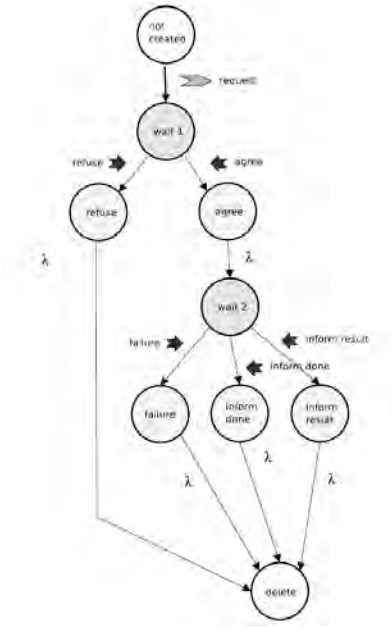


Fig. 5. Finite State Machine for FIPA-request in Magentix

stop them according to its needs. The conversation manager is an abstraction that hides the basic concepts of the conversations (makes sure the message is exchanged, mailbox management, etc.) from the agent developer, which only has to specify what to do in each step of the protocol, easily allowing the concurrent execution and management of several conversations. We are now working to extend the conversation management funcionalities. We especially want to facilitate the specification of any protocol interaction that agent developer could require, apart from that predefined by FIPA.

6. The Tourism Service Application

In this section, we present a real application developed in Magentix which uses some of the features provided. In order to test the performance of a MAP focused on large systems, we require examples aimed to be large-scale which are so real as possible. The Tourism Service application [39] is a MAS that allows users to find information about places of interest in a city according to their preferences (restaurants, movie theaters, museums, theaters, and other places of general interest such as monuments, churches, beaches, parks, etc.), by using their mobile phone or PDA. Once a specific place has been selected, the tourist can make a reservation at a restaurant, buy tickets for a film, etc. Our

research group has been working with a partnership developing MAS-based recommender systems for tourists.

There are four different agent types in the application. A SightAgent manages all of the information related to the features and activities for a specific place of interest in the city. A UserAgent allows tourists to interact with the system by means of a GUI on their mobile devices. A BrokerAgent mediates between UserAgents and SightAgents. It also manages updated information about the SightAgents registered on it. Finally, a PlanAgent manages all of the planning processes in the system. The application offers search, reservation, planning, and registration services. The Search service is offered by the BrokerAgent and can be requested by a UserAgent. The result of the invocation of this service is a list of descriptions of places that match user preferences. The Reserve service is offered by a SightAgent and can be requested by a UserAgent. The result of this service is the confirmation of a successful reservation or an error message. The ”Plan a Specific Day” service is provided by the PlanAgent and can be requested by a UserAgent. The result of this service is a plan consisting of a list of places or activities.

We have implemented this application using the Magentix MAP with RDF support. The implemented ontology is represented in RDF and gives detailed descriptions of tourist places, information about scheduling, etc; For example, information about restaurants, represent issues related to menus, cuisine, ingredients, etc.

UserAgents can be implemented as Magentix agents in the MAP or by means of an interface that is implemented using the J2ME (Java 2 Micro Edition) specification. In the latter case, UserAgents have to make HTTP requests to a GatewayAgent, which acts as a gateway between UserAgents and the rest of the system. This GatewayAgent is implemented as a Magentix agent, which includes a micro-http server. This mechanism allows the interaction between Magentix agents and external agents.

7. Large Scale Evaluation of the Messaging Service

In this section, we present different experiments in order to evaluate the messaging service of Magentix, based on the application presented in Section 6. As we stated in Section 2, this service is crucial when developing systems with large agent populations with high message traffic. In [16], we presented a testbed for MAP performance evaluation. These tests focused on evaluating different parameters of the MAP in one and two hosts: the message traffic, the message size, the registered services, the searched services, the CPU consumption of the threads, the memory consumption, the network traffic, etc. According to these tests, the main bottleneck of a MAP performance is related to the messaging service. These conclusions have also been confirmed by other authors, who claim that other parameters such as the CPU cycles do not reach saturations in large-scale environments [28]. Based on these conclusions, in [49] we presented a set of large-scale benchmarks to test the messaging ser-

vice. The experiments shown here are based on these benchmarks and are adapted to the Tourism Service Application presented in Section 6.

We compare Magentix against the performance of Jade, which is a well-known MAP and is more scalable than other MAPs as we stated in Section 2. Since the initial implementation of the Tourism Service Application was in Jade [38], we can determine the performance of the messaging service of both MAPs simulating different scenarios in this domain. We used 20 PCs Intel(R) Core(TM) 2 Duo CPU @ 2.60GHz, 2GB RAM, Ubuntu 10.10 and Linux Kernel 2.6.35. The computers were connected to each other via a 100Mb Ethernet hub.

The first experiment is aimed at testing the MAPs performance when both the number of agents and the message traffic increase. This experiment measures the capability of the MAP when messages are sent to different agents. As an example, this situation can occur when a BrokerAgent requests different SightAgents. We simulate this scenario by launching several groups of BrokerAgents and SightAgents. The objective of each BrokerAgent is to send a message to the first SightAgent on its list, which sends back the same message. After that, each BrokerAgent sends a message to its corresponding SightAgent placed in the next host and waits for the response. This experiment measures the time elapsed between when the first message is sent by the first BrokerAgent and when the last message is received by the last BrokerAgent. The experiment started with 100 agents in the system, increasing to 1000. The number of messages sent by each BrokerAgent was specified at 1000.

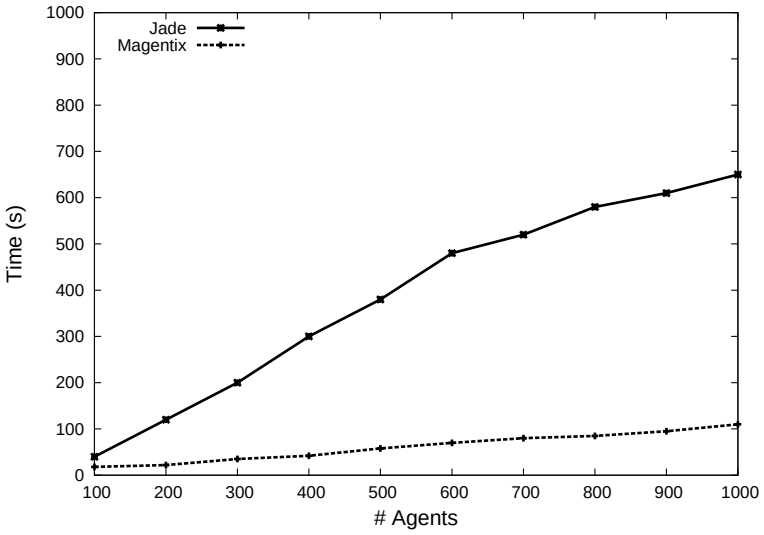


Fig. 6. Experiment 1: population and traffic increase

Figure 6 shows the time required for the two MAPs. The figure shows that there is a performance degradation as the number of agents and the message traffic increase. However, Magentix performance degrades less than Jade performance. As an example, it can be observed that the elapsed time in Magentix when the system is composed of 1000 agents is less than the elapsed time in Jade when the system is composed of 200 agents.

Another typical scenario is the massive amount of message-sending to a specific agent. The second experiment measures the ability of the MAPs when a lot of agents send messages to a single one. This specific agent could become a bottleneck in the system when multiple messages are addressed to it. This scenario appears, for example, when UserAgents are requesting the same BrokerAgent to retrieve information. The BrokerAgent has to serve every received request. As the number of incoming requests increases, the time for processing these requests may also increase. In order to simulate this, a single BrokerAgent agent and several UserAgents were launched. The goal of each UserAgent was to send messages to the BrokerAgent. The elapsed time between when the BrokerAgent received the first message and when it answered all the messages is shown in Figure 7. In this experiment, we increased the number of UserAgents up to 100, distributed among all the hosts. Each UserAgent sent 10000 messages.

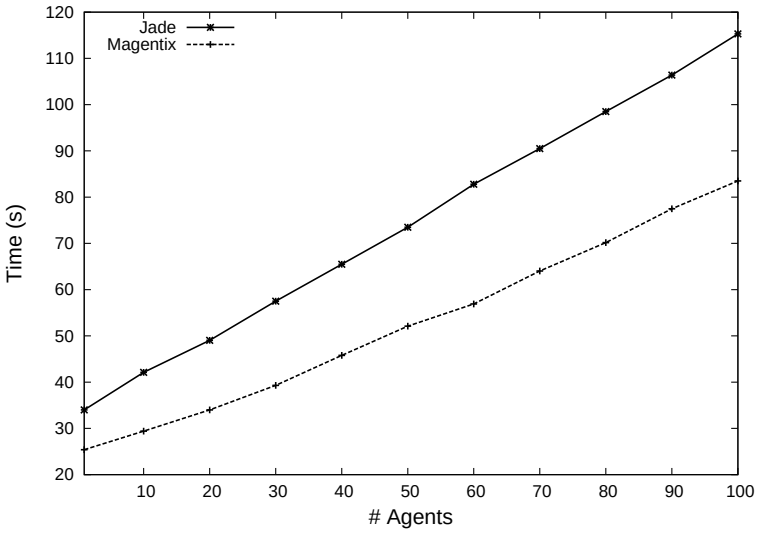


Fig. 7. Experiment 2: massive sending to an agent

It can be observed that the elapsed time increases in both MAPs as the number of requests increases. However, as in the first experiment, the performance degradation is less in Magentix. The time difference between the two

MAPs gradually increases as the number of agents increases. Therefore, Magentix is also more scalable and efficient than Jade in this scenario. Note that in this scenario the receiver agent is not changed during the entire experiment.

The third experiment complements the second one. The distribution of agents in this experiment was similar. However, there were the same number of BrokerAgents as UserAgents. In this experiment, several BrokerAgents were placed in the same host and each UserAgent communicated with its corresponding BrokerAgent. The results obtained are shown in Figure 8. It can be observed that the results for Jade are similar to the results for the second experiment. This is due to the way that Jade implements communication among all the MAP hosts. Therefore, the bottleneck is caused by the message transport system and not by the way the message queue is managed by the agent itself. In contrast, the performance in Magentix in the third experiment is slightly better than in the second one.

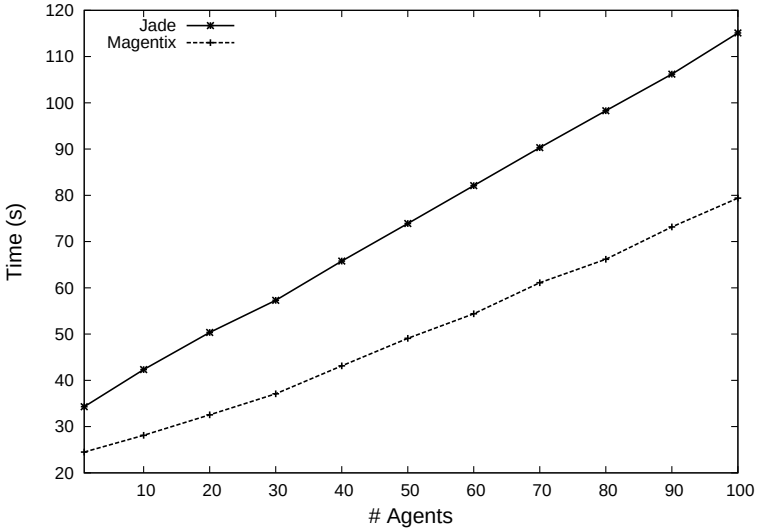


Fig. 8. Experiment 3: host massive sending

The fourth experiment checks the limits of the MAPs. This experiment provides a different perspective from the previous experiments in which the receiver agents are predefined. This may give rise to different bottlenecks, showing another typical scenario in real systems, in which some agents may be more requested than others. In order to simulate this, several BrokerAgents were placed in 10 hosts of the MAP and several UserAgents were placed in the other 10 hosts. Each UserAgent had to send 1000 messages to a non-predefined BrokerAgent. Thus, the specific BrokerAgent was randomly selected before sending each message. This caused some BrokerAgents to be more

overloaded than others. Furthermore, in this experiment, the number of agents was increased to 2000, in order to overload the MAPs.

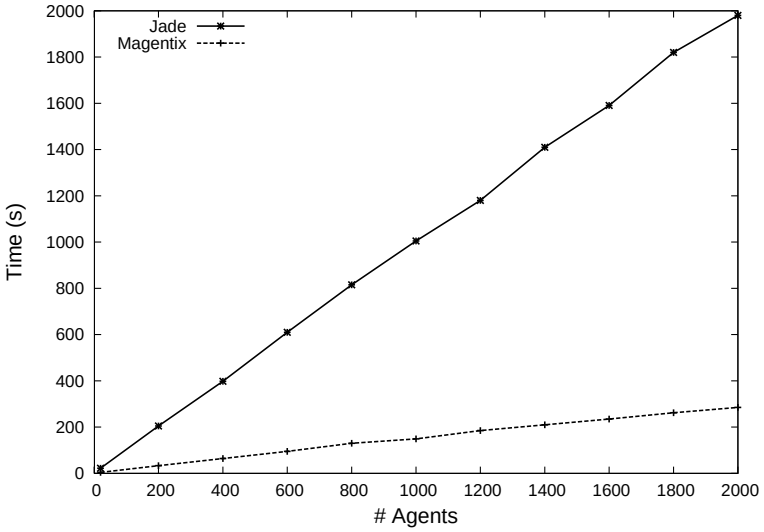


Fig. 9. Experiment 4: random requests

It can be observed in Figure 9 that Magentix offers better performance than Jade, and the differences increase according to the increase in the traffic. The figure also shows that the two MAPs present higher response times with respect to the first experiment, in which the traffic was equally distributed among all the BrokerAgents. This is due to the fact that, in this forth experiment, message load is not spread over all of the receiver agents launched. Since the BrokerAgent in each message sending is selected randomly, there may be BrokerAgents that have to serve a lot of messages while others are idle. Therefore, as the second experiment indicates, Jade performs quite badly when there is an agent that is receiving a lot of messages. As a result, performance differences with respect to the first experiment are much higher in Jade than in Magentix.

From the results provided in these tests, we can conclude that Magentix improves the efficiency and scalability of the messaging service provided by Jade, which is the most commonly used MAP and that it is more scalable than other MAPs. In these tests, we have simulated four typical scenarios in order to determine the efficiency and scalability in the Magentix and the Jade MAPs. These tests represent critical situations so that we can see the degree of performance improvement achieved more clearly. Although we scale up to 20 hosts in these tests, the conclusions obtained can be extended to at least to 100 hosts according to the results shown in [49].

8. Conclusions

The next generation of technologies aims to provide features such as distribution, interoperability, scalability, organizations, service-oriented, open, geographically dispersed, and so on. MASs can contribute to these environments by evolving new applications that will become more autonomous and social from the point of view of the MAS field.

MAPs have traditionally been used as a support framework to facilitate the development of these kinds of systems. A lot of MAPs have been developed in the last few years; however, unfortunately, very few real MAS-based applications have appeared, probably due to the lack of suitability of the support frameworks which did not fulfill all of the requirements. In order to support the new generation of systems (in line with the latest trends in rapidly expanding technologies), new MAP designs should focus on being interoperable, scalable, and large-scale as just some of their key features.

In this paper, we have presented the Magentix MAP. Since its design is closer to the OS level, it ensures that the MAP is efficient, especially when running large systems. Basic services such as an agent directory service, a service directory service, and a messaging service are provided by Magentix. We have implemented and tested the performance of this MAP. Magentix also provides a group-oriented communication mechanism. This mechanism allows communication between individual agents as well as interaction among groups of agents. When considering large systems, security concerns become an important issue and a necessary feature when these systems become open. Magentix has a security model that is based on the Kerberos protocol and Linux OS access control which provides authentication, integrity, and confidentiality. In order to achieve interoperable systems, we represented the information using RDF. This framework has been widely used in MAS for different purposes. Magentix represents messages to be exchanged in RDF so that agents can easily manage the information that is sent and received. Ontologies defined in OWL have also used to interact with services.

Using a tourism service application, we have shown how Magentix can be used as a support framework to develop MAS-based applications. The messaging service evaluation shown in this paper demonstrates that a MAP design that uses the OS services provides greater efficiency and scalability than other high-performance middleware-based MAPs such as Jade.

With the features provided by Magentix we can establish the next objective of the project: to provide Magentix with support for open MAS. We are working on the development of an http-based gateway at MAP level, in order to allow the interaction between Magentix agents and agents developed in other MAPs. Virtual organizations where agents dynamically enter and exit the system and form groups could also be created in Magentix.

Acknowledgments. This work has been partially supported by CONSOLIDER-INGENIO 2010 under grant CSD2007-00022, and projects TIN2011-27652-C03-01 and TIN2008-

04446. Juan M. Alberola has received a grant from Ministerio de Ciencia e Innovación de España (AP2007-00289).

References

1. Fipa-os. http://fipa-os.sourceforge.net
2. FIPA (The Foundation for Intelligent Physical Agents). http://www.fipa.org/
3. Jack. http://www.agent-software.com
4. Madkit. http://www.madkit.org
5. OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features/
6. RDF. http://www.w3.org/TR/rdf-primer/
7. RDF/XML Syntax Specification. http://www.w3.org/TR/rdf-syntax-grammar/
8. Redland RDF Libraries. http://librdf.org
9. Safeguard. http://www.ist-safeguard.org/
10. Standard for information technology - portable operating system interface (POSIX)
11. Tryllian agent development kit (adk). http://www.tryllian.com
12. Zeus agent toolkit. http://labs.bt.com/projects/agents/zeus/
13. SACI - simple agent communication infrastructure. http://www.lti.pcs.usp.br/saci/ (2009)
14. Alberola, J.M., Mulet, L., Such, J.M., Garcia-Fornes, A., Espinosa, A., Botti, V.: Operating system aware multiagent platform design. In: Proceedings of the Fifth European Workshop on Multi-Agent Systems (EUMAS-2007). pp. 658–667 (2007)
15. Alberola, J.M., Such, J.M., Espinosa, A., Botti, V., Garcia-Fornes, A.: Scalable and efficient multiagent platform closer to the operating system. Artificial Intelligence Research and Development 184, 7–15 (2008)
16. Alberola, J.M., Such, J.M., Garcia-Fornes, A., Espinosa, A., Botti, V.: A performance evaluation of three multiagent platforms. In: Artificial Intelligence Review, Volume 34, Number 2. pp. 145–176 (2010)
17. Batouma, N., Sourrouille, J.L.: Dynamic adaption of resource aware distributed applications. In: International journal of grid and distributed computing. vol. 4, pp. 25–42 (2011)
18. Bauwens, B.: Xml-based agent communication: Vpn provisioning as a case study. In: XML Europe'99 (1999)
19. Bellifemine, F., Caire, G., Poggi, A., Rimassa, G.: Jade a white paper. EXP 3, 6–19 (2003)
20. Bitting, E., C.J.G.A.: Multiagent system development kit: An evaluation. In: Proceedings of Communication Networks and Services Research Conference, May 15-16, pp. 80-92, Moncton, New Brunswick, Canada, 2003
21. Bădică, C., Budimac, Z., Burkhard, H.D., Ivanovic, M.: Software Agents: Languages, Tools, Platforms. Computer Science and Information Systems 8(2), 255–298 (2011)
22. Burbeck, K., Garpe, D., Nadjm-Tehrani, S.: Scale-up and performance studies of three agent platforms. In: IPCCC 2004 (2004)
23. Camacho, D., Aler, R., Castro, C., Molina, J.M.: Performance evaluation of zeus, jade, and skeletonagent frameworks. In: IEEE International Conference on Systems, Man and Cybernetics, 2002 (2002)
24. Cenk, R., Dikenelli, O., Seylan, I., Gürcan, Ö.: An infrastructure for the semantic integration of fipa compliant agent platforms. In: AAMAS. pp. 1316–1317 (2004)
25. Chmiel, K., T.D.G.M.K.P.: Testing the efficency of jade agent platform. In: Proceedings of the ISPDC/HeteroPar'04, 49-56 (2004)

26. Cortese, E., F.Quarta, Vitaglione, G.: Scalability and performance of jade message transport system. EXP 3, 52–65 (2003)
27. E. Argente, A. Gilet, S.V.V.J., Botti, V.: Survey of mas methods and platforms focusing on organizational concepts. In: Recent advances in Artificial Intelligence Research and Development. vol. 113, pp. 309–316. IOS Press (2004)
28. Fernández, V., Grimaldo, F., Lozano, M., Orduña, J.M.: Evaluating jason for distributed crowd simulations. In: ICAART (2). pp. 206–211 (2010)
29. FIPA: FIPA Abstract Architecture Specification. FIPA (2001), http://www.fipa.org/specs/fipa00001/
30. FIPA: FIPA ACL Message Structure Specification. FIPA (2001), http://www.fipa.org/specs/fipa00061/
31. FIPA: FIPA RDF Content Language Specification. FIPA (2001), http://www.fipa.org/specs/fipa00011/
32. FIPA: FIPA Interaction Protocol Library Specification. FIPA (2003), http://www.fipa.org/specs/fipa00025/
33. Giang, N.T., Tung, D.T.: Agent platform evaluation and comparison (2002)
34. Hirsch, B., Konnerth, T., Heßler, A.: Merging agents and services — the JIAC agent platform. In: Multi-Agent Programming: Languages, Tools and Applications, pp. 159–185. Springer (2009)
35. Huynh, D., Karger, D.R., Quan, D.: Haystack: A platform for creating, organizing and visualizing information using rdf. In: Eleventh World Wide Web Conference Semantic Web Workshop (2002)
36. Laclavik, M., Balogh, Z., Gatial, E., Hluchy, L.: Agent architecture based on semantic knowledge model. In: 5th annual conference. VSB-Technick. pp. 288–291 (2006)
37. Lee, L.C., Ndumu, D.T., Wilde, P.D.: The stability, scalability and performance of multi-agent systems. BT Technology Journal 16, 94–103 (1998)
38. Lopez, J.S., Bustos, F.A., Julian, V., Rebollo, M.: Developing a Multiagent Recommender System: A Case Study in Tourism Industry. International Transactions on Systems Science and Applications 4(3), 206–212 (2008)
39. Lopez, J.S., Bustos, F.A., Inglada, V.J.: Tourism services using agent technology: A multiagent approach. INFOCOMP - Journal of Computer Science - Special Edition pp. 51–57 (2007)
40. Lynch, S.: Using meta-agents to build mas platforms and middleware. In: International Conference on Agents and Artificial Intelligence (ICAART) (2011)
41. Mulet, L., Such, J.M., Alberola, J.M.: Performance evaluation of open-source multiagent platforms. In: Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS06). pp. 1107–1109. Association for Computing Machinery, Inc. (ACM Press) (2006)
42. Nodine, M.H., Unruh, A.: Facilitating open communication in agent systems: the infosleuth infrastructure (1997)
43. Omicini, A., Rimassa, G.: Towards seamless agent middleware. In: TAPOC 2004
44. Park, A.H., et al.: A flexible and scalable agent platform for multi-agent systems. In: Proceedings of WASET Bangkok (2007) (2007)
45. Pesovic, D., Vidakovic, M., Ivanovic, M., Budimac, Z., Vidakovic, J.: Usage of agents in document management. Computer Science and Information Systems 8(1), 193–210 (2011)
46. Shakshuki, E.: A methodology for evaluating agent toolkits. In: ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume I. pp. 391–396. IEEE Computer Society, Washington, DC, USA (2005)

47. Such, J.M., Alberola, J.M., Espinosa, A., Garca-Fornes, A.: A Group-oriented Secure Multiagent Platform. Software: Practice and Experience 41(11), 1289–1302 (2011)
48. Such, J.M., Alberola, J.M., Garca-Fornes, A., Espinosa, A., Botti, V.: Kerberos-based secure multiagent platform. In: Sixth International Workshop on Programming Multi-Agent Systems (ProMAS'08). pp. 173–186 (2008)
49. Such, J.M., Alberola, J.M., Mulet, L., Espinosa, A., Garcia-Fornes, A., Botti, V.: Large-scale multiagent platform benchmarks. In: LAnguages, methodologies and Development tools for multi-agent systemS (LADS 2007). Proceedings of the Multi-Agent Logics, Languages, and Organisations - Federated Workshops. pp. 192–204 (2007)
50. Such, J.M., Alberola, J.M., Mulet, L., Garcia-Fornes, A., Espinosa, A., Botti, V.: Hacia el diseo de plataformas multiagente cercanas al sistema operativo. In: International workwhop on practical applications on agents and multi-agent systems (2007)
51. Vrba, P.: Java-based agent platform evaluation. In: Proceedings of the HoloMAS 2003. pp. 47–58 (2003)

Juan M. Alberola is a PhD student at the Departament de Sistemes Informàtics i Computació of the Universitat Politècnica de València. His interest areas include agent organizations, adaptation, multiagent platforms, case-based-reasoning and electronic markets.

Jose M. Such is Lecturer in the School of Computing and Communications at Lancaster University (UK). He was previously research fellow at Universitat Politècnica de València (Spain), by which he was awarded a PhD in Computer Science in 2011. He is mostly interested in the following research topics: Privacy, Security, Trust, Reputation, Multi-agent Systems, and Artificial Intelligence.

Vicent Botti is Full Professor at the Universitat Politècnica de València (Spain) and head of the GTI-IA research group of the Departament de Sistemes Informàtics i Computació. He received his Ph.D. in Computer Science from the same university in 1990. His research interests are multi-agent systems, agreement technologies, and articial intelligence, where he has more than 200 refereed publications in international journals and conferences. Currently he is Vice-rector of the Universitat Politècnica de València.

Agustín Espinosa is Lecturer at the Departament de Sistemes Informàtics i Computació of the Universitat Politècnica de València and a researcher at the GTI-IA Research Group of the Universitat Politècnica de València. His research interests include multiagent systems, agent architectures, agent platforms, agent frameworks, and real-time agents. He received his Ph.D. in Computer Science from the Universitat Politècnica de València, Spain in 2003.

Ana García-Fornes is a Professor at the Departament de Sistemes Informàtics i Computació of the Universitat Politècnica de València. Her interest areas include: real-time artificial intelligence, real-time systems, development of multiagent infrastructures, tracing systems, operating systems based on agents, agent organizations, and negotiation strategies.

Received: October 29, 2011; Accepted: October 8, 2012.

DOI:10.2298-CSIS120713042R

Validation of Schema Mappings with Nested Queries

Guillem Rull, Carles Farré, Ernest Teniente, and Toni Urpí

Departament d'Enginyeria de Serveis i Sistemes d'Informació
Universitat Politècnica de Catalunya (UPC)—BarcelonaTech
1-3 Jordi Girona, 08034 Barcelona, Spain
{grull, farre, teniente, urpi}@essi.upc.edu

Abstract. With the emergence of the Web and the wide use of XML for representing data, the ability to map not only flat relational but also nested data has become crucial. The design of schema mappings is a semi-automatic process. A human designer is needed to guide the process, choose among mapping candidates, and successively refine the mapping. The designer needs a way to figure out whether the mapping is what was intended. Our approach to mapping validation allows the designer to check whether the mapping satisfies certain desirable properties. In this paper, we focus on the validation of mappings between nested relational schemas, in which the mapping assertions are either inclusions or equalities of nested queries. We focus on the nested relational setting since most XML's Document Type Definitions (DTDs) can be represented in this model. We perform the validation by reasoning on the schemas and mapping definition. In particular, we encode the given mapping scenario into a single flat database schema, and reformulate each desirable property check as a query satisfiability problem.

Keywords: schema mapping, nested relational model, nested query, query equality, query inclusion, validation.

1. Introduction

Schema mappings are specifications that model a relationship between two data schemas. They are key elements in any system that requires the interaction of heterogeneous data and applications [16]. Such interaction usually involves databases that have been independently developed and that store the data of the common domain under different representations; that is, the involved databases have different schemas. In order to make the interaction possible, schema mappings are required to indicate how the data stored in each database relates to the data stored in the other databases. This problem, known as *information integration*, has been recognized as a challenge faced by all major organizations, including enterprises and governments [5].

With the emergence of the Web and the wide use of XML for representing data, the ability to map not only flat relational but also nested data has become crucial. A sign of this is the growing interest of the research community during the last years on the topics of XML mappings—see, for instance, [3, 4]—and mappings between nested relational schemas—e.g., [20, 15].

However, the mapping design process is not a fully automatic one. A human designer is needed to guide the process, choose among mapping candidates, and successively refine the mapping [20]. Intricate manual work may actually be required to refine a particular mapping. Since manual design is labor-intensive and error-prone, the designer needs a way to figure out whether the mapping is what was intended.

In order to address this need of validation, we propose an approach that allows the designer to ask questions about the mapping. In particular, it allows the designer to check whether the mapping satisfies certain desirable properties. In this paper, we focus on two properties that have been identified as important properties of mappings in the literature: *mapping satisfiability* [3] and *mapping inference* [19]. An additional property, *mapping losslessness* [22], is also addressed in the extended version of the paper [23].

Our approach is based on reasoning on the schemas and the mapping definition, and does not rely on specific schema instances, since that might not reveal all the potential pitfalls.

In this paper, we focus on the application of this validation approach to mapping scenarios in which nested data is involved. More specifically, we address the validation of mapping scenarios in which the source and the target schema are nested relational [20], and in which the mapping is a set of assertions. Mapping assertions are in the form of either query inclusions, i.e., $Q_S \subseteq Q_T$, or query equalities, i.e., $Q_S = Q_T$, where Q_S and Q_T are queries over the source and the target schema, respectively, and whose result is a nested relation (i.e., Q_S and Q_T are *nested queries*). Note that a query inclusion (equality) assertion holds for a given pair of mapped schema instances if and only if the answer to Q_S over the source instance is a subset of (equal to) the answer to Q_T over the target instance.

We focus on the nested relational setting since it covers the most common class of the well-known Document Type Definitions (DTDs) [3], and also because it is the model that is typically used in the data exchange context to represent semi-structured schemas [20].

The class of schemas and mappings that we consider is quite expressive. We consider schemas with integrity constraints, where these constraints are in the form of disjunctive embedded dependencies [10] (this class of dependencies is applied here to the nested relational setting instead of the traditional flat relational one in the same way as tuple-generating dependencies are applied to the nested relational setting in [20]). The integrity constraints of the schemas and the queries of the mapping may contain arithmetic comparisons and negations. Union of nested queries is also allowed. This class of mapping scenarios subsumes those considered by previous works on mapping validation [7, 6, 1], which also focus on the nested

relational setting but do not consider arithmetic comparisons nor negation. Moreover, these previous works deal with a class of constraints and mapping assertions—in the form of tuple-generating dependencies [13]—that is known to be a particular class of the disjunctive embedded dependencies that we consider [10].

To actually perform the validation, we propose a reformulation of each desirable property check in terms of the query satisfiability problem over a single flat relational database. Given a nested relational mapping scenario, we encode it into a flat database and define a query over this database such that the query is satisfiable if and only if the desirable property holds. This encoding takes into account the nested structure of the schemas, their integrity constraints, and the nested queries defined over them. Moreover, this encoding rewrites the mapping assertions as integrity constraints over the new flat relational database.

In this way, we extend our previous work on validating relational mappings [22] and make it applicable to the nested case.

We solve the query satisfiability problem by means of the Constructive Query Containment (CQC) method [14]. This method is able to deal with flat relational databases in which queries and integrity constraints have no recursion and may contain safe negation—on base and derived predicates—, equality and inequality ($\neq$) comparisons, and also order comparisons ($<$, $\leq$, $>$, $\geq$). To the best of our knowledge, the CQC method is the only query satisfiability method able to handle this class of schemas and queries. The use of the this method together with the encoding that we present in this paper is what allows us to address nested relational mapping scenarios that are more expressive than the ones addressed in the previous literature.

Reasoning on the class of mapping scenarios that we consider here is, unfortunately, undecidable. However, extending the approach proposed by [21], we studied in [24] a series of conditions that, if satisfied, guarantee the termination of the CQC method for the current query satisfiability check. A detailed performance evaluation of the CQC method has been done in [22, 24] for the case of flat relational mapping scenarios. This performance evaluation showed that, for those scenarios in which termination is guaranteed, the cost of the method is exponential with respect to the size of the mapping scenario, as expected given the complexity of reasoning on such an expressive language.

We would also like to remark that the reduction that we propose of each desirable property in terms of query satisfiability is linear with respect to the size of the given mapping scenario. Moreover, this reduction does not increase the complexity of the problem, that is, checking query satisfiability is not more complex than checking the desirable properties [22].

We have performed some experiments to show the feasibility of our approach, using mapping scenarios from the *STBenchmark* [2]. The results are reported in the extended version of the paper [23].

Summarizing, the main contributions of the paper are the following:

- We validate nested relational mappings by means of checking whether they satisfy certain desirable properties. We focus on two properties that have

been identified as important properties of mappings: mapping satisfiability and mapping inference.
- We consider a class of mapping scenarios that is significantly more expressive than those considered by previous works on nested relational mapping validation.
- We propose an encoding of the nested relational schemas in the mapping scenario into a single flat relational database.
- We propose a rewriting of the mapping assertions as integrity constraints over the new relational database.
- We extend our previous work on validating relational mappings [22] to the nested relational case. In particular, we propose a reformulation of each desirable property of nested relational mappings in terms of the query satisfiability problem over a flat relational database. Such a query satisfiability check can be solved by means of the CQC method.

To better motivate the kind of validation that we propose, the next subsection discusses detailed examples. The rest of the paper is structured as follows. Section 2 introduces base concepts. Section 3 outlines our approach for validating mappings with nested queries. Section 4 and Section 5 detail how to encode a given nested relational mapping scenario into a single flat database schema. Section 6 explains how to reformulate the check of each desirable property of mappings in terms of the query satisfiability problem. Section 7 reviews the related work. Section 8 concludes the paper.

1.1 Examples of Mapping Validation

Consider a mapping scenario in which an airline company wants to publish information about their flights and connecting flights into a certain flight-searching Web site. Fig. 1 shows the source and the target schema of this scenario, where dashed lines denote referential constraints and the underlined attribute denotes a key.

Example 1
Let us assume the mapping designer has come up with two mapping

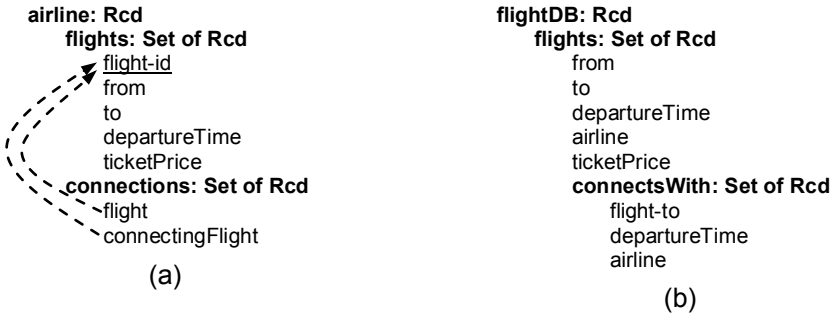


Fig. 1. *Example source (a) and target (b) nested relational schemas.*

candidates. The first candidate is a mapping with two assertions: {m_1, m_2}. Assertion m_1 maps the information of individual flights available in the source schema, independently of whether these flights have connecting flights or not. Assertion m_2, maps the information about the connecting flights.

m_1:

```
for f in airline.flights
return f.from, f.to, f.departureTime,
        f.ticketPrice, "airlineXY"
```

$\subseteq$

```
for f in flightDB.flights
return f.from, f.to,
        f.departureTime,
        f.ticketPrice, f.airline
```

m_2:

```
for  c in airline.connections,
     f1 in airline.flights,
     f2 in airline.flights
where c.flight = f1.flight-id and
      c.connectingFlight = f2.flight-id
return  f1.from, f1.to,
f1.departureTime,
        "airlineXY", f1.to,
        f1.departureTime, "airlineXY"
```

$\subseteq$

```
for  f in flightDB.flights,
     c in f.connectsWith
return f.from, f.to,
f.departureTime,
        f.airline, c.flight-to,
        c.departureTime, c.airline
```

The second candidate is a mapping with a single assertion: {m_3}. Assertion m_3 maps both the information of individual flights and of their connecting flights at the same time. It uses nested queries to ensure that flights without connecting flights are also mapped; that is, for each flight in the source, it creates a tuple that contains not only the flight's data but also a set with the corresponding connecting flights; a set that may be empty if the flight has no connecting flights.

m_3:

```
for  f in airline.flights
return  f.from, f.to, f.departureTime,
        "airlineXY", f.ticketPrice,
        for  c in airline.connections,
             f2 in airline.flights
        where c.flight = f.flight-id
           and c.connectingFlight =
                f2.flight-id
        return f2.to, f2.departureTime,
                "airlineXY"
```

$\subseteq$

```
for  f in flightDB.flights
return  f.from, f.to, f.departureTime,
        f.airline, f.ticketPrice,
        for  c in f.connectsWith
        return c.flight-to,
                c.departureTime,
                c.airline
```

The designer could think that both mapping candidates may be actually equivalent and that in that case he would feel more inclined to choose mapping {m_3} since it seems more compact. Let us suppose that the designer wants to be sure before making the decision. He could then check whether m_3 is actually inferred from {m_1, m_2}, and whether m_1 and m_2 are both inferred from {m_3}.

The check of the *mapping inference* property [19] would reveal that while assertions m_1 and m_2 are indeed inferred from mapping {m_3}, assertion m_3 is not inferred from mapping {m_1, m_2}. Fig. 2 shows an instantiation of the mapping scenario that exemplifies the latter, i.e., it shows a source and a target instance that satisfy {m_1, m_2} but not m_3. The example shows that mapping {m_1, m_2} does not ensure the correlation between a flight's ticket price and the flight's connecting flights. Notice that there is one single flight

(a) Source instance:

flights				
flight-id	**from**	**to**	**departureTime**	**ticketPrice**
1	A	B	T_1	50
2	A	C	T_2	70
3	C	D	T_3	45
4	C	E	T_4	60
5	C	F	T_5	55

connections	
flight	**connectingFlight**
2	3
2	4
2	5

(b) Target instance:

flights							
from	**to**	**departureTime**	**airline**	**ticketPrice**	**connectsWith**		
A	B	T_1	airlineXY	50	∅		
A	C	T_2	airlineXY	70	∅		
C	D	T_3	airlineXY	45	∅		
C	E	T_4	airlineXY	60	∅		
C	F	T_5	airlineXY	55	∅		
A	C	T_2	airlineXY	80	**flight-to**	**departureTime**	**airline**
					D	T_3	airlineXY
					E	T_4	airlineXY
A	C	T_2	airlineXY	90	**flight-to**	**departureTime**	**airline**
					F	T_5	airlineXY

Fig. 2. *Example (a) source and (b) target instances.*

with connecting flights on the source instance, and that the data of that flight is split in three tuples on the target instance: a first one with no connecting flights but with the right ticket price, a second one with a wrong ticket price and with only two of the three connecting flights, and a third one also with a wrong ticket price and with the remaining connecting flight.

The designer could thus conclude that mapping $\{m_3\}$ is preferable not only because is more compact but also because is more accurate than $\{m_1, m_2\}$.

Example 2

Let us assume now that, according to a new business rule, only the most expensive connecting flights should be advertised by means of the flight-searching Web site. Let us also assume that the Web site has a constraint t_1 according to which, only flights with a ticket price no greater than 200 can be published.

t_1: for f in flightDB.flights then f.ticketPrice $\leq$ 200

Taking into account the business requirement and target schema's t_1 constraint, the designer could decide to adapt mapping $\{m_3\}$ and introduce and additional condition in the inner query block (shown in bold). The result is mapping $\{m_4\}$.

Let us also assume that another business rule was introduced, which the designer thinks has no effect on the mapping. The requirement is enforced by a new constraint s_1 on the source schema, which requires that the connecting flights must be cheaper than the initial flight.

s_1: for c in airline.connections, f_1 in airline.flights, f_2 in airline.flights
where c.flight = f_1.flight-id and c.connectingFlight = f_2.flight-id
then f_2.ticketPrice < f_1.ticketPrice

m_4:

```
for f in airline.flights
return f.from, f.to, f.departureTime,
       "airlineXY", f.ticketPrice,
       for c in airline.connections,
           f2 in airline.flights
       where c.flight = f.flight-id
          and c.connectingFlight =
              f2.flight-id
          and f2.ticketPrice = 200
       return f2.to, f2.departureTime,
              "airlineXY"
```

$\subseteq$

```
for f in flightDB.flights
return f.from, f.to, f.departureTime,
       f.airline, f.ticketPrice,
       for c in f.connectsWith
       return c.flight-to,
              c.departureTime,
              c.airline
```

In order to be sure that no further modifications to the mapping should be made as a result of this new business requirement, the designer could check the non-trivial *satisfiability* of mapping $\{m_4\}$ at all its levels of nesting. By doing that, he would realize that m_4's inner level of nesting never maps any data, i.e., mapping $\{m_4\}$ is only mapping those flights with no connecting flights. The problem is that there is a contradiction between source constraint s_1 and the source query of m_4; in particular, since the source query of m_4 selects only connecting flights with ticket price equal to 200 in its inner query block, and s_1 requires these connecting flights to be cheaper than the initial flight selected by the outer query block, that implies the initial flight should have a ticket price greater than 200, which is not allowed by the target schema.

2. Preliminaries

In this section, we introduce the basic concepts of nested relational mapping scenarios and of flat relational databases. We also discuss the query satisfiability problem and its solution by means of the CQC method.

2.1 Nested Relational Mapping Scenarios

A *nested relation* $R(A1, ..., A_n)$ is a relation in which each attribute A_i can be defined either as a simple type (e.g., integer, real, string) or as another nested relation. For instance, the nested relation *flights* on Fig. 1b has five simple-type attributes: *from*, *to*, *departureTime*, *airline* and *ticketPrice*; and one attribute that is also a nested relation: *connectsWith*.

A *nested relational schema* consists of a root record whose elements are either simple types or nested relations. Nested relational model generalizes the relational one. A flat relational schema can be modeled as a nested relational schema in which the root record is a collection of flat relations, i.e., relations with all their attributes defined as simple types. Fig. 1a shows a flat relational schema and Fig. 1b a truly nested relational one.

We consider nested relational schemas with integrity constraints. An *integrity constraint* is a Boolean condition in the form (we adapt the XQuery-like notation of [20]):

for $variable_1$ in $relation_1$, ..., $variable_n$ in $relation_n$ where $condition_1$ then $condition_2$

The variables in the for clause are bound to tuples from the relation that follows the in. A $variable_i$ can be used in $relation_{i+1}$,...,$relation_n$, $condition_1$ and $condition_2$. The condition in the where and then clauses denotes a Boolean expression that may include arithmetic comparisons (=, $\neq$, <, $\leq$, >, $\geq$) and make use of conjunction, disjunction, and negation. As an example, see the constraints s_1 and t_1 on the Example 2 of Section 1.1.

An *instance* of a nested relational schema is *consistent* if it satisfies all the integrity constraints defined over the schema. Fig. 2 shows a consistent instance for each of the two schemas in Fig. 1.

A *nested query* is a query whose answer is a nested relation. That is, nested queries define derived nested relations. We use a notation similar to that of the integrity constraints (also adapted from [20]):

for $variable_1$ in $relation_1$, ..., $variable_n$ in $relation_n$
where $condition_1$ return $result_1$, ..., $result_n$

where each $result_i$ can be either a simple-type expression or another nested query. See, for example, the queries on assertion m_3 in the Example 1 of Section 1.1.

A *mapping scenario* is a triplet (S, T, M), where S is a source nested relational schema, T is a target nested relational schema, and M is a set of mapping assertions.

A mapping assertion m is a pair of nested queries related by a $\subseteq$ or = operator; the query on the left-hand side being defined over the source schema, and the query on the right-hand side being defined over the target schema: $Q_{source} \subseteq / = Q_{target}$.

An instantiation of a mapping scenario (S, T, M) consists of an instance I_S of S and an instance I_T of T, such that I_S and I_T satisfy all the assertions in M.

A mapping assertion $Q_{source} \subseteq / = Q_{target}$ is satisfied by instances I_S, I_T iff the answer to Q_{source} on I_S is included/equal to the answer to Q_{target} on I_T.

We apply the definition of inclusion and equality of nested relations used in [18].

The *inclusion* of two nested structures R_1, R_2 of the same type T, i.e., $R_1 \subseteq R_2$, can be defined by induction on T as follows:

(1) If T is a simple type, $R_1 \subseteq R_2$ iff $R_1 = R_2$
(2) If T is a record type (i.e., a tuple), $R_1=[R_{1,1},\ldots,R_{1,n}] \subseteq R_2=[R_{2,1},\ldots,R_{2,n}]$ iff $R_{1,1} \subseteq R_{2,1} \wedge \ldots \wedge R_{1,n} \subseteq R_{2,n}$
(3) If T is a set type, $R_1=\{R_{1,1},\ldots,R_{1,n}\} \subseteq R_2=\{R_{2,1},\ldots,R_{2,n}\}$ iff $\forall i\, \exists j\, R_{1,i} \subseteq R_{2,j}$

Equality of nested structures, i.e., $R_1 = R_2$, can be defined similarly:

(1) If T is a simple type, $R_1 = R_2$
(2) If T is a record type, $[R_{1,1},\ldots,R_{1,n}] = [R_{2,1},\ldots,R_{2,n}]$ iff $R_{1,1} = R_{2,1} \wedge \ldots \wedge R_{1,n} = R_{2,n}$

(3) If T is a set type, $\{R_{1,1},\ldots,R_{1,n}\}=\{R_{2,1},\ldots,R_{2,n}\}$ iff $\forall i\ \exists j\ R_{1,i} = R_{2,j} \wedge \forall j\ \exists i\ R_{2,j} = R_{1,i}$

Note that, given the definitions above, $Q_1 = Q_2$ is not equivalent to $Q_1 \subseteq Q_2 \wedge Q_2 \subseteq Q_1$ [18].

2.2 Flat Relational Databases

A flat relational schema is a finite set of flat relations with integrity constraints. We use first-order logic notation and represent relations by means of predicates. Each predicate P has a *predicate definition* $P(A1,\ldots,A_n)$, where $A1,\ldots,A_n$ are the *attributes*. A predicate is said to be of *arity n* if it has n attributes. Predicates may be either *base predicates*, i.e., the tables in the database, or *derived predicates*, i.e., queries and views. Each derived predicate Q has attached a set of non-recursive deductive rules that describe how Q is computed from the other predicates. A *deductive rule* has the following form:

$$q(\bar{X}) \leftarrow r_1(\bar{Y}_1) \wedge \ldots \wedge r_n(\bar{Y}_n) \wedge \neg r_{n+1}(\bar{Z}_1) \wedge \ldots \wedge \neg r_m(\bar{Z}_s) \wedge C_1 \wedge \ldots \wedge C_t$$

Each C_i is a *built-in literal*, that is, a literal in the form of $t_1\ op\ t_2$, where $op \in \{<, \leq, >, \geq, =, \neq\}$ and t_1 and t_2 are terms. A *term* can be either a variable or a constant. Literals $r_i(\bar{Y}_i)$ and $\neg r_i(\bar{Z}_i)$ are positive and negated *ordinary literals*, respectively (note that in both cases r_i can be either a base predicate or a derived predicate). Literal $q(\bar{X})$ is the *head* of the deductive rule, and the other literals are the *body*. Symbols $\bar{X}$, $\bar{Y}_i$ and $\bar{Z}_i$ denote lists of terms. We assume deductive rules to be *safe*, which means that the variables in $\bar{Z}_i$, $\bar{X}$ and C_i are taken from $\bar{Y}_1$, …, $\bar{Y}_n$, i.e., the variables in the negated literals, the head and the built-in literals must appear in the positive literals in the body. Literals about base predicates are often referred to as *base literals* and literals about derived predicates are referred to as *derived literals*.

We consider integrity constraints that are *disjunctive embedded dependencies* (DEDs) [10] extended with arithmetic comparisons and the possibility of being defined over views (i.e., they may have derived predicates in their definition). A *constraint* has one of the following two forms:

$$r_1(\bar{Y}_1) \wedge \ldots \wedge r_n(\bar{Y}_n) \rightarrow C_1 \vee \ldots \vee C_t$$

$$r_1(\bar{Y}_1) \wedge \ldots \wedge r_n(\bar{Y}_n) \wedge C_1 \wedge \ldots \wedge C_t \rightarrow \exists \bar{V}_1\ r_{n+1}(\bar{U}_1) \vee \ldots \vee \exists \bar{V}_s\ r_{n+s}(\bar{U}_s)$$

Each $\bar{V}_i$ is a list of fresh variables (i.e., variables that have not been used anywhere else before), and the variables in $\bar{U}_i$ are taken from $\bar{V}_i$ and $\bar{Y}_1$, ..., $\bar{Y}_n$. Note that each predicate r_i (on both sides of the implication) can be either base or derived. We refer to the left-hand side of a constraint as the *premise*, and to the right-hand side as the *consequent*.

Formally, we write $S = (PD, DR, IC)$ to indicate that S is a database schema with predicate definitions PD, deductive rules DR, and integrity constraints IC. We omit the PD component when it is clear from the context.

An *instance D* of a schema *S* is a set of facts about the base predicates of *S*. A *fact* is a *ground literal*, i.e., a literal with all its terms constant. An instance *D* is *consistent* with schema *S* if it satisfies all the constraints in *IC*. The extension of the queries and views of *S* when evaluated on *D* is the *intensional database* (IDB) of *D*, denoted IDB(*D*). The answer to a query *Q* on an instance *D*, denoted $A_Q(D)$, is the set of all facts about predicate *q* in the IDB of *D*, i.e., $A_Q(D) = \{q(\bar{a}) \mid q(\bar{a}) \in \text{IDB}(D)\}$, where $\bar{a}$ denotes a list of constants.

2.3 Query Satisfiability and the CQC Method

A query *Q* is said to be *satisfiable* on a database schema *S* if there is some consistent instance *D* of *S* in which *Q* has a non-empty answer, i.e., $A_Q(D) \neq \varnothing$ [17].

The *CQC* (*Constructive Query Containment*) *method* [14], originally designed to check query containment, tries to build a consistent instance of a database schema in order to satisfy a given goal (a conjunction of literals). Clearly, using literal $q(\bar{X})$ as goal, where $\bar{X}$ is a list of distinct variables, results in the CQC method checking the satisfiability of query *Q*.

The CQC method starts by taking the empty instance and uses different *Variable Instantiation Patterns* (VIPs) based on the syntactic properties of the views/queries and constraints in the schema, attempting to generate only the relevant facts that are to be added to the instance under construction. If the method is able to build an instance that satisfies all the literals in the goal and does not violate any of the constraints, then that instance is a solution and proves the goal is satisfiable. The key point is that the VIPs guarantee that if the variables in the goal are instantiated using the constants they provide and the method does not find any solution, then no solution is possible.

The solution space that the CQC method explores is a tree, called the *CQC-tree*. Each branch of the CQC-tree is what is called a *CQC-derivation*. A CQC-derivation can be either *finite* or *infinite*. Finite CQC-derivations can be either *successful*, if they reach a solution, or *failed*, if they reach a violation that cannot be repaired. As proven in [14], the CQC method terminates when there is no solution, that is, when all CQC-derivations are finite and failed, or when there is some finite solution, i.e., when there is a finite, successful CQC-derivation.

A series of sufficient conditions for the termination of the CQC method has been studied in [24]. These conditions extend the ones proposed by [21].

A detailed performance evaluation of the CQC method has been done in [22, 24] for the case of flat relational mapping scenarios. It showed that, for those scenarios in which termination is guaranteed, the cost of the method is exponential with respect to the size of the mapping scenario. This is expected given the complexity of reasoning on such an expressive class of mapping scenarios.

3. Validation by Means of Checking Desirable Properties

We understand mapping validation as checking whether the mapping being designed meets the intended needs and requirements. To perform this validation, we propose to allow the designer to check whether the mapping has certain desirable properties. In this paper, we focus on two desirable properties of mappings (we will provide the formal definition of these properties in Section 6): satisfiability and inference.

As illustrated in the Example 2 of Section 1.1, mapping satisfiability allows detecting contradictions either between the mapping assertions or between the mapping assertions and the integrity constraints of the schemas. Mapping inference allows to detect redundancies in the mapping, i.e., redundant mapping assertions, and also to compare mapping candidates.

In order to actually check these desirable properties of mappings, we propose to translate the mapping scenario from the nested relational setting into the flat relational one. That implies flattening not only the nested relational schemas, but also the nested queries on the mappings. Then, we propose to take advantage of previous work on validating mappings in the relational setting [22] and reformulate the desirable property checking on this new flat relational mapping scenario in terms of the query satisfiability problem. To do so, we firstly combine the relational versions of the two mapped schemas into a single relational schema. Secondly, we rewrite the mapping assertions as integrity constraints over this single relational schema. Finally, for each mapping desirable property that we want to check on the original nested mapping scenario, we define a query on the single relational schema in such a way that this query will be satisfiable on this schema if and only if the mapping desirable property holds on the original mapping scenario.

In the next sections we discuss in detail how to translate the nested relational mapping scenario into the flat relational one (Section 4 and Section 5), and how to reformulate each desirable property check in terms of a query satisfiability problem over this flat relational translation (Section 6).

4. Flattening Nested Schemas and Queries

In this section, we detail how to encode the nested schemas and the nested queries of a given nested relational mapping scenario into a single flat database schema. Note that when we say nested queries we mean those in the mapping assertions. We will later rely on this encoding of the nested queries to rewrite the mapping assertions as integrity constraints.

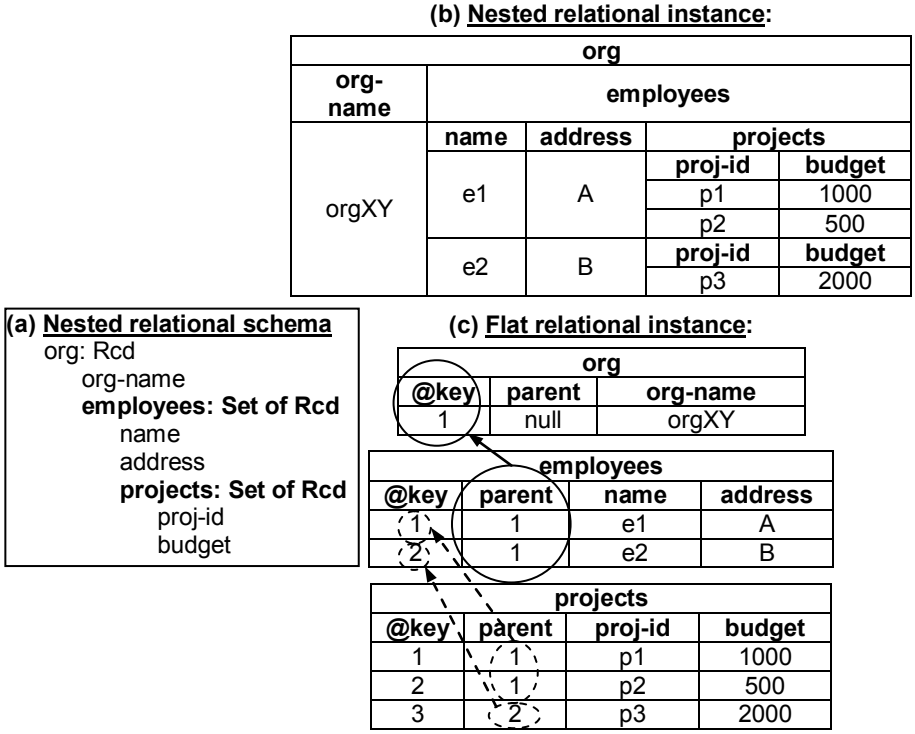


Fig. 3. *A (a) nested relational schema, an (b) instance of this schema, and (c) the translation of the instance into flat relations*

4.1 Nested Schemas

Our translation of nested relational schemas into flat relational ones is based on the *hierarchical representation* used by Yu and Jagadish in [25]. They address the problem of discovering functional dependencies on nested relational schemas. They translate the schemas into a flat representation, so algorithms for finding functional dependencies on relational schemas can be applied.

The hierarchical representation assigns a flat relation to each nested table. To illustrate that, consider the nested relational schema in Fig. 3a, which models data about an organization, its employees, and the projects each employee works on. The hierarchical representation, as defined in [25], of this nested relational schema would be the following set of flat relations:

{org(@*key*, *parent*, *org-name*), employees(@*key*, *parent*, *name*, *address*),
projects(@*key*, *parent*, *proj-id*, *budget*)}

Note that each flat relation keeps the simple-type attributes of the nested relation, and has two additional attributes: the *@key* attribute, which models an implicit tuple id; and the *parent* attribute, which references the *@key* attribute of the parent table and models the parent-child relationship of the nested relations. Fig. 3b shows an instance of the previous nested relational schema, and Fig. 3c shows the corresponding instance of the flat relational schema into which the previous nested schema is translated.

For simplicity, we skip the flat relation of the root record when it has only set-type attributes and no simple-type ones. We also skip the *parent* attribute of those relations that do not have a parent relation, and we skip the *@key* attribute of those relations that do not have child relations. For example, we would translate the source and target schema of the mapping scenario in Fig. 1 into flat relations as follows:

$$\textit{source} = \{\text{flights}_S(\textit{flight-id}, \textit{from}, \textit{to}, \textit{departureTime}, \textit{ticketPrice}),\ \text{connections}(\textit{flight}, \textit{connectingFlight})\}$$

$$\textit{target} = \{\text{flights}_T(@\textit{key}, \textit{from}, \textit{to}, \textit{departureTime}, \textit{airline}, \textit{ticketPrice}),\ \text{connectsWith}(\textit{parent}, \textit{flight-to}, \textit{departureTime}, \textit{airline})\}$$

The semantics of the *@key* and *parent* attributes are made explicit by means of adding the corresponding key and referential constraints to the flat relational schema that results from the flattening process. As an example, the flat version of the target schema in Fig. 1 (see above) would include the following key and referential constraint:

key: $\text{flights}_T(@\text{key}, f, t, dt, a, tp) \wedge \text{flights}_T(@\text{key}', f, t, dt, a, tp) \rightarrow @\text{key} = @\text{key}'$
ref: $\text{connectsWith}(\text{parent}, ft, dt, a) \rightarrow \text{flights}_T(\text{parent}, f', t', dt', a', tp')$

The integrity constraints that already exist on the original nested schemas can be straightforwardly translated into constraints over the flat relational version of the schema. For example, let us consider again the source and target constraint s_1 and t_1 of the Example 2 of Section 1.1; the constraints would be translated into the following:

s_1': $\text{connections}(f, cf) \wedge \text{flights}_S(f, frm, to, dt, tp) \wedge \text{flights}_S(cf, frm', to', dt', tp') \rightarrow tp' < tp$
t_1': $\text{flights}_T(@k, frm, to, dt, a, tp) \rightarrow tp \leq 200$

4.2 Nested Queries

Regarding the flattening of nested queries, we follow a variation of the approach used in [18]. In this approach, each nested query is translated into a collection of flat queries, one for each nested query block. For example, let us consider the source schema from Fig. 1, and let us suppose that we have the following nested query *Q* defined over this source schema, which selects the flights with a ticket price of at least 50 and, for each of these flights, selects the connecting flights that are cheaper than the original flight:

```
Q: for f in airline.flights where f.tp ≥ 50
   return f.from, f.to, f.departureTime, "airlineXY", f.ticketPrice,
          for c in airline.connections, f2 in airline.flights
          where c.flight = f.flight-id and c.connectingFlight = f2.flight-id
                and f2.ticketPrice < f.ticketPrice
          return f2.to, f2.departureTime, "airlineXY"
```

The nested query *Q* has two query blocks: the outer block Q_{outer}

```
Qouter: for f in airline.flights where f.tp ≥ 50
        return f.from, f.to, f.departureTime, "airlineXY", f.ticketPrice
```

and the inner block Q_{inner}.

```
Qinner: for c in airline.connections, f2 in airline.flights
        where c.flight = f.flight-id and c.connectingFlight = f2.flight-id
              and f2.ticketPrice < f.ticketPrice
        return f2.to, f2.departureTime, "airlineXY"
```

Since both query blocks are flat queries when considered independently, and assuming we have already flattened the corresponding schema (the source schema in this case), each of these blocks can be straightforwardly translated into a query over the flat version of the schema. The only technical detail, and the main difference with respect to [18], is the treatment of the *inherited variables*—called *indexes* in [18]—, which are the variables defined in the for clause of the outer block that are also used in the inner block. In [18], the translation of both the outer and the inner block would be extended to select the key attributes of the relations bound to the inherited variables; in the case of the inner block, since it uses the inherited variables but does not define them, that would require copying in the inner block's translation those literals from the outer block's translation that correspond to the definition of the inherited variables. In our example, the inherited variable "f" is defined in the translation of Q_{outer} by the literal "flights_S(fid, frm, to, dt, tp)", where "fid" corresponds to the key attribute and is selected by this translation. The translation of Q_{inner} should thus contain a copy of this literal (shown below in bold) and also select "fid":

$$Q_{outer}(\text{fid}, \text{frm}, \text{to}, \text{dt}, \text{"airlineXY"}) \leftarrow \text{flights}_S(\text{fid}, \text{frm}, \text{to}, \text{dt}, \text{tp}) \wedge \text{tp} \geq 50$$

$$Q_{inner}(\text{fid}, \text{to}', \text{dp}', \text{"airlineXY"}) \leftarrow \text{connections}(\text{fid}, \text{cf}) \wedge \text{flight}_S(\text{cf}, \text{frm}', \text{to}', \text{dp}', \text{tp}') \wedge \mathbf{\text{flights}_S(\text{fid}, \text{frm}, \text{to}, \text{dt}, \text{tp})} \wedge \text{tp}' < \text{tp}$$

Notice that without the literal in bold, Q_{inner} would not have access to variable "tp" (i.e., f.ticketPrice) and could not make the required comparison.

The flat relational equivalent to answering the original nested query *Q* would be making a left outer join of the translations of Q_{outer} and Q_{inner} with "fid" as the join variable.

In order to simplify the translation, not only that of the nested queries themselves but specially the translation of the mapping assertions (see next section), we use access patterns [9]; in particular, we consider derived relations with "input-only" attributes in addition to the traditional "input-output" ones. We use $R\langle x_1, ..., x_n\rangle(y_1, ..., y_n)$ to denote that $x_1,...,x_n$ are input-only

terms bound to derived relation R, and $y_1,...,y_n$ are input-output ones. As an example, we would translate Q_{inner} and Q_{outer} as follows:

Q_{outer}(**fid**, **tp**, frm, to, dt, "airlineXY") ← $flights_S$(**fid**, frm, to, dt, **tp**) ∧ **tp** ≥ 50

Q_{inner}<**fid**, **tp**>(to′, dp′, "airlineXY") ← connections(**fid**, cf) ∧ $flight_S$(cf, frm′, to′, dp′, tp′) ∧ tp′ < **tp**

Notice that we enforce the translation of Q_{outer} to select the variables "fid" and "tp", which are then to be inherited by Q_{inner} through its input-only attributes. Note also that there is no need now to repeat the ordinary literal of Q_{outer} in Q_{inner}.

In order for a deductive rule to be *safe*, the variables that appear as input-only terms of some literal in the body of the rule must either appear as input-output terms of some other positive ordinary literal in the same body, or appear in the head of the rule as input-only terms. Similarly, the variables that appear in a negated or built-in literal in the body of a rule must either appear as input-output terms of some other positive ordinary literal in the same body, or appear in the head of the rule as input-only terms. See for instance, variable "tp" in Q_{inner} above, which appears in the body of the rule in a built-in literal, and in the head of the rule as an input-only term.

5. Rewriting Mapping Assertions As Integrity Constraints

A nested relational mapping scenario consists of two nested relational schemas and a mapping with nested queries that relates them. We have already discussed how to translate each nested schema into the flat relational formalism. In order to complete the translation of the nested relational mapping scenario into the flat relational setting, we must see now how to translate the mapping assertions. We assume the queries in both sides of the mapping are part of each schema's definition and have already been translated along with them.

To translate a mapping assertion $Q_{source} \subseteq / = Q_{target}$, we will make use of the definition of inclusion/equality of nested structures from [18] (see Section 2.1), and we will rely on the flat queries that result from flattening Q_{source} and Q_{target}. As an example, consider the mapping assertion m_3 from Example 1 (see Section 1.1). Let us assume the source and target query—let us call them Q^S and Q^T—are translated into the flat queries Q^S_{outer}, Q^S_{inner} and Q^T_{outer}, Q^T_{inner}, respectively, as follows:

Q^S_{outer}(fid, frm, to, dt, "airlineXY", tp) ← $flights_S$(fid, frm, to, dt, tp)
Q^S_{inner}<fid>(to, dt, "airlineXY") ← connections(fid, cf) ∧ $flights_S$(cf, frm, to, dt, tp)

Q^T_{outer}(@k, frm, to, dt, a, tp) ← $flights_T$(@k, frm, to, dt, a, tp)
Q^T_{inner}<@k>(to, dt, a) ← connectsWith(@k, to, dt, a)

According to the semantics of query inclusion, two schema instances I_S and I_T satisfy m_3 if and only if the answer to Q^S on I_S, i.e., $AQ^S(I_S)$, is included in the answer to Q^T on I_T, i.e., $AQ^T(I_T)$. Recall that, as defined in [18], a nested

structure such as $AQ^S(I_S)$ is included in another nested structure such as $AQ^T(I_T)$ if and only if each tuple *a* in $AQ^S(I_S)$ "matches" some tuple *b* of $AQ^T(I_T)$, where "matches" means that each simple-type attribute on *b* (e.g., the *from* attribute) must have the same value than the corresponding attribute of *a*, and that the value of each set-type attribute on *b* (e.g., the *connectsWith* attribute) must be a set that recursively includes the set bound to the corresponding set-type attribute of *a*. Notice that this is a recursive definition, where simple-type attributes are the base case and set-type ones are the recursive case.

We can express the above definition as a Boolean condition over the flat translations of the mapped schemas. The condition will be true if the given schema instances satisfy the mapping assertion, and false otherwise. The condition begins with the requirement that for all tuple *a* returned by the outer query block of Q^S there must be a matching *b* on the result of the outer query block of Q^T with the same value on the simple-type attributes:

$$\forall \text{fid,frm,to,dt,a,tp } (Q^S_{\text{outer}}(\text{fid, frm , to, dt, a, tp}) \rightarrow \exists @k \ (Q^T_{\text{outer}}(@k, \text{frm, to, dt, a, tp}) \ldots$$

The condition must also include the requirement that the set-type attributes of *a* must be included in the corresponding set-type attributes of *b*, i.e., the recursive case:

$$\ldots \wedge \forall to', dt', a' \ (Q^S_{\text{inner}}\langle \text{fid} \rangle(to', dt', a') \rightarrow Q^T_{\text{inner}}\langle @k \rangle(to', dt', a')) \ ))$$

By making the union of the flat mapped schemas and introducing this Boolean condition as an integrity constraints over this union, we will get that each consistent instance of the resulting flat database schema will correspond to a consistent instantiation of the mapping scenario (i.e., an instantiation in which the mapping assertions are true), and vice versa. The only problem is that the above condition does not fit the syntactic requirements of the class of constraints we consider, i.e., disjunctive embedded dependencies (DEDs), which are expressions of the form $\forall \overline{X} \ (\phi(\overline{X}) \rightarrow \exists \overline{Y}_1 \ \psi_1(\overline{X}, \ \overline{Y}_1) \vee \ldots \vee \exists \overline{Y}_n \ \psi_n(\overline{X}, \ \overline{Y}_n))$ in which $\forall$ quantifiers are not allowed inside $\psi_1,\ldots,\psi_n$. Fortunately, we can take advantage of the fact that we are able to deal with negation and get rid of that inner $\forall$ quantifier. We can introduce a double negation $\neg\neg$ in front of the $\forall$ quantifier, and move one of the negations inwards:

$$\ldots \wedge \neg\exists to', dt', a' \ (Q^S_{\text{inner}}\langle \text{fid} \rangle(to', dt', a') \wedge \neg Q^T_{\text{inner}}\langle @k \rangle(to', dt', a')) \ ))$$

There are only two details remaining now. The first is that we only allow direct negation of single literals and not of conjunction of literals. However, we do allow negation of derived literals, so we can just fold the conjunction into a new derived relation:

$$\forall \text{fid,frm,to,dt,a,tp } (Q^S_{\text{outer}}(\text{fid, frm , to, dt, a, tp}) \rightarrow \exists @k \ (Q^T_{\text{outer}}(@k, \text{frm, to, dt, a, tp}) \wedge \neg \mathbf{Q^S_{\text{inner}}\text{-not-included-in-}Q^T_{\text{inner}}\langle \text{fid, } @k \rangle()} \ ))$$

where

$$Q^S_{\text{inner}}\text{-not-included-in-}Q^T_{\text{inner}}\langle \text{fid, } @k \rangle() \leftarrow Q^S_{\text{inner}}\langle \text{fid} \rangle(to', dt', a') \wedge \neg Q^T_{\text{inner}}\langle @k \rangle(to', dt', a')$$

The second detail is that we do not allow the explicit use of negation in the integrity constraints, i.e., the literals in ϕ and in $\psi_1,...,\psi_n$ cannot be negated. We do however allow constraints in which the consequent is a contradiction, e.g., 1 = 0. With that and the introduction of double negation in front of the remaining $\forall$ quantifier, we can rewrite the expression as follows. First, we introduce the double negation and move one of the negation inwards just as we did before:

$$\neg\exists \text{fid,frm,to,dt,a,tp } (Q^S_{outer}(\text{fid, frm, to, dt, a, tp}) \wedge \neg\exists @k\ (Q^T_{outer}(@k, \text{frm, to, dt, a, tp}) \wedge \neg \mathbf{Q^S_{inner}\text{-not-included-in-}Q^T_{inner}<fid, @k>()}\ ))$$

To get rid of the inner $\neg\exists$ quantifier, we fold the conjunction into a new derived relation:

$$\neg\exists \text{fid,frm,to,dt,a,tp } (Q^S_{outer}(\text{fid, frm, to, dt, a, tp}) \wedge \neg \mathbf{aux\text{-}\ Q^S_{outer}\text{-not-included-in-}Q^T_{outer}<fid, frm, to, dt, a, tp>()}$$

where

$$\text{aux- } Q^S_{outer}\text{-not-included-in-}Q^T_{outer}<\text{fid, frm, to, dt, a, tp}>() \leftarrow Q^T_{outer}(@k, \text{frm, to, dt, a, tp}) \wedge \neg Q^S_{inner}\text{-not-included-in-}\ Q^T_{inner}<\text{fid, @k}>()$$

We still make an additional folding to get rid of the remaining $\neg\exists$ quantifier, and we get:

$$\neg \mathbf{Q^S_{outer}\text{-not-included-in-}Q^T_{outer}()}$$

where

$$Q^S_{outer}\text{-not-included-in-}Q^T_{outer}() \leftarrow Q^S_{outer}(\text{fid, frm, to, dt, a, tp}) \wedge \neg\text{aux- } Q^S_{outer}\text{-not-included-in-}Q^T_{outer}<\text{fid, frm, to, dt, a, tp}>()$$

Finally, we can get rid of the $\neg$ by stating that the atom implies a contradiction:

$$Q^S_{outer}\text{-not-included-in-}Q^T_{outer}() \rightarrow \mathbf{1 = 0}$$

This constraint, together with the deductive rules that define the new derived relations that we just introduced, enforces the mapping assertion m_3.

In a more generic way, the rewriting of a query inclusion mapping assertion can be formalized as follows.

Let Q^A and Q^B be two generic (sub)queries with compatible answer:

Q^A:for var_1 in rel_1, ..., var_{na} in rel_{na} where $cond$ return A_1, ..., A_m, B_1, ..., B_k
Q^B:for var_1' in rel_1',..., var_{nb}' in rel_{nb}' where $cond'$ return A_1', ..., A_m', B_1',..., B_k'

where each A_i and A_i' are simple-type expressions, and each B_i and B_i' are subqueries. Let us assume the outer block of Q^A is translated into the derived relation $Q^A_{outer}\langle x_1, ..., x_{ka}\rangle(v_1, ..., v_{na}, r_1, ..., r_m)$, where $x_1,...,x_{ka}$ denote the variables inherited from the ancestor query blocks, $v_1,...,v_n$ denote the additional variables to be inherited by the inner query blocks of Q^A_{outer}, and $r_1,...,r_m$ denote the simple-type values returned by the block. Similarly, let us also assume the outer block of Q^B is translated into $Q^B_{outer}\langle x_1', ..., x_{kb}'\rangle(v_1', ..., v_{nb}', r_1', ..., r_m')$.

We use T-inclusion(Q^A, Q^B, $\{i_1, \ldots, i_h\}$) to denote the translation of $Q^A \subseteq Q^B$, where $\{i_1, \ldots, i_h\}$ is the union of the variables inherited by Q^A and Q^B from their respective parent blocks (if any):

$$\text{T-inclusion}(Q^A, Q^B, \{i_1,...,i_h\}) = \neg Q^A\text{-}not\text{-}included\text{-}in\text{-}Q^B\langle i_1,...,i_h\rangle$$

where

$$Q^A\text{-}not\text{-}included\text{-}in\text{-}Q^B\langle i_1,...,i_h\rangle \leftarrow Q^A{}_{outer}\langle x_1,...,x_{ka}\rangle(v_1,...,v_{na}, r_1,...,r_m) \wedge \neg aux\text{-}Q^A\text{-}not\text{-}included\text{-}in\text{-}Q^B\langle i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m\rangle$$

$$aux\text{-}Q^A\text{-}not\text{-}included\text{-}in\text{-}Q^B\langle i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m\rangle \leftarrow Q^B{}_{outer}\langle x_1',...,x_{kb}'\rangle(v_1',...,v_{nb}', r_1,...,r_m) \wedge \text{T-inclusion}(B_1, B_1', \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\}) \wedge ... \wedge \text{T-inclusion}(B_k, B_k', \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\})$$

If Q^A and Q^B are not subqueries but full queries, then the following constraint is to be introduced:

$$\neg\text{T-inclusion}(Q^A, Q^B, \{i_1,...,i_h\}) \rightarrow 1 = 0$$

Similarly, the rewriting of a generic query equality assertion $Q^A = Q^B$ as a set of integrity constraints can be formalized as follows:

$$\neg\text{T-equality}(Q^A, Q^B, \{i_1, \ldots, i_h\}) \rightarrow 1 = 0$$
$$\neg\text{T-equality}(Q^B, Q^A, \{i_1, \ldots, i_h\}) \rightarrow 1 = 0$$

where

$$\text{T-equality}(Q^A, Q^B, \{i_1,...,i_h\}) = \neg Q^A\text{-}not\text{-}eq\text{-}to\text{-}Q^B\langle i_1,...,i_h\rangle$$

and

$$Q^A\text{-}not\text{-}eq\text{-}to\text{-}Q^B\langle i_1,...,i_h\rangle \leftarrow Q^A{}_{outer}\langle x_1,...,x_{ka}\rangle(v_1,...,v_{na}, r_1,...,r_m) \wedge \neg aux\text{-}Q^A\text{-}not\text{-}eq\text{-}to\text{-}Q^B\langle i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m\rangle$$

$$aux\text{-}Q^A\text{-}not\text{-}eq\text{-}to\text{-}Q^B\langle i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m\rangle \leftarrow Q^B{}_{outer}\langle x_1',...,x_{kb}'\rangle(v_1',...,v_{nb}', r_1,...,r_m) \wedge \text{T-equality}(B_1, B_1', \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\}) \wedge \text{T-equality}(B_1', B_1, \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\}) \wedge ... \wedge \text{T-equality}(B_k, B_k', \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\}) \wedge \text{T-equality}(B_k', B_k, \{i_1,...,i_h, v_1,...,v_{na}, r_1,...,r_m, v_1',...,v_{nb}'\})$$

The two constraints above, together with the deductive rules of the corresponding derived relations, enforce the definition of query equality as defined in [18] (see Section 2.1).

Intuitively, T-equality(Q^A, Q^B, $\{i_1,\ldots,i_h\}$) denotes the condition that, for each instantiation of the mapping scenario, each tuple in the answer to Q^A must have an equal tuple in the answer to Q^B. Notice that in order to fully express the definition of query equality, we need to enforce both T-equality(Q^A, Q^B, $\{i_1,\ldots,i_h\}$) and T-equality(Q^B, Q^A, $\{i_1,\ldots,i_h\}$).

6. Desirable Properties in Terms of Query Satisfiability

In this section, we show how two desirable properties of mappings—satisfiability and inference—can be reformulated as a query satisfiability check over the flat relational translation of mapping scenarios we have presented in Section 4 and Section 5.

6.1 Mapping Satisfiability

We say a mapping is *satisfiable* if there is a pair of schema instances that make all the mapping assertions true in a non-trivial way. An example of trivial satisfaction would be a pair of empty schema instances, which is not the case we are interested in here. We distinguish two kinds of satisfiability: *strong* and *weak*.

Intuitively, a mapping is strongly satisfiable if all its mapping assertions can be non-trivially satisfied at the same time at all their levels of nesting, e.g., the inner query block of mapping assertion m_4's source query from the Example 2 of Section 1.1 never maps any data (i.e., always provides an empty answer); therefore, although the outer query block does map some data, mapping $\{m_4\}$ is not strongly satisfiable.

Definition 1 (*Strong Satisfiability*). A mapping M is *strongly satisfiable* iff there exist I_S, I_T instances of the source and target schema, respectively, such that I_S and I_T satisfy the assertions in M, and for each assertion Q_{source} *op* Q_{target} in M, the answer to Q_{source} in I_S is a strong answer. We say R is a *strong answer* iff

(1) R is a simple type value,
(2) R is a record $[R_1, ..., R_n]$ and $R_1, ..., R_n$ are all strong answers, or
(3) R is a non-empty set $\{R_1, ..., R_n\}$ and $R_1, ..., R_n$ are all strong answers.

Intuitively, we say a mapping is weakly satisfiable if at least one mapping assertion can be satisfied at least at its outermost level of nesting. As an example, mapping $\{m_4\}$ is indeed weakly satisfiable.

Definition 2 (*Weak satisfiability*). A mapping M is *weakly satisfiable* iff there exist I_S, I_T instances of the source and target schema, respectively, and some mapping assertion m: $Q_{source} \subseteq/= Q_{target}$ in M, such that I_S, I_T make m true and the answer to Q_{source} on I_S is not empty, i.e., $A_{Q_{source}}(I_S) \neq \emptyset$.

Let us assume M is a mapping with assertions $\{Q^S_1 \; op \; Q^T_1, ..., Q^S_n \; op \; Q^T_n\}$. Let $S = (PD_S, DR_S, IC_S)$ be the flat translation of the source schema, and $T = (PD_T, DR_T, IC_T)$ be the flat translation of the target schema. Let us also assume that IC_M and DR_M are the constraints and deductive rules that result from the rewriting of the assertions in M. The flat database schema that encodes the mapping scenario is:

$$DB = (PD_S \cup PD_T, \; DR_S \cup DR_T \cup DR_M, \; IC_S \cup IC_T \cup IC_M)$$

The reformulation of strong satisfiability of M as a query satisfiability check over DB is the following:

$$Q_{strongSat} \leftarrow \text{StrongSat}(Q^S_1, \varnothing) \wedge ... \wedge \text{StrongSat}(Q^S_n, \varnothing)$$

where StrongSat is a function generically defined as follows. Let Q be a generic (sub)query:

$$Q\text{: for } var_1 \text{ in } rel_1, ..., var_s \text{ in } rel_s \text{ where } cond \text{ return } A_1, ..., A_m, B_1, ..., B_k$$

where $A_1,...,A_m$ are simple-type expressions and $B_1,...,B_k$ are inner query blocks. Let predicate Q_{outer} be the translation of the outer query block of Q. Then,

$$\begin{aligned}\text{StrongSat}(Q, inheritedVars) = {} & Q_{outer}\langle x_1,...,x_r\rangle(v_1,...,v_s, r_1,...,r_m) \wedge \\ & \text{StrongSat}(B_1, inheritedVars \cup \{v_1,...,v_s, r_1,...,r_m\}) \wedge ... \wedge \\ & \text{StrongSat}(B_k, inheritedVars \cup \{v_1,...,v_s, r_1,...,r_m\})\end{aligned}$$

where $\{x_1,...,x_r\} \subseteq inheritedVars$.

Boolean query $Q_{strongSat}$ is satisfiable over DB if and only if mapping M is strongly satisfiable.

Intuitively, if we can find an instance of DB that satisfies $Q_{strongSat}$, we can obtain from that database instance a source and a target instance for the mapping scenario. These two instances will be consistent with their respective schemas and with the mapping assertions because DB includes the corresponding integrity constraints. The strong satisfiability property will hold, because $Q_{strongSat}$ is encoding its definition.

As an example, let us assume the outer query block of mapping assertion m_4's source query in Example 2 is translated into derived relation Q^S_{outer}(*flight-id*, *from*, *to*, *departureTime*, *airline*, *ticketPrice*), and the inner query block into derived relation Q^S_{inner}<*flight-id*>(*to*, *departureTime*, *airline*). Then, strong satisfiability of $\{m_4\}$ would be reformulated as follows:

$$Q_{strongSat} \leftarrow Q^S_{outer}(fid, frm, to, dt, a, tp) \wedge Q^S_{inner}\text{<fid>}(to', dt', a')$$

The reformulation of weak satisfiability of M as a query satisfiability check over DB is the following:

$$\begin{aligned} Q_{weakSat} &\leftarrow Q^S_{1,outer}(\bar{X}_1) \\ &\cdots \\ Q_{weakSat} &\leftarrow Q^S_{n,outer}(\bar{X}_n) \end{aligned}$$

where $Q^S_{1,outer},...,Q^S_{n,outer}$ are the translations of the outermost query blocks of the source mapping's queries.

Boolean query $Q_{weakSat}$ is satisfiable over DB if and only if mapping M is weakly satisfiable.

The intuition is that $Q_{weakSat}$ can only be if some of the outermost blocks of the source mapping's queries is not empty. Therefore, if $Q_{weakSat}$ is true, we can extract from the corresponding instance of DB an instantiation of the mapping scenario that exemplifies the property.

As an example, weak satisfiability of mapping {m_4} would be reformulated as follows:

$$Q_{weakSat} \leftarrow Q^{S}_{outer}(fid, frm, to, dt, a, tp)$$

Notice that there is only one deductive rule for $Q_{weakSat}$ because the mapping has only one assertion.

6.2 Mapping Inference

The *mapping inference* property [19] checks whether a given mapping assertion is inferred from a set of others assertions. It can be used, for instance, to detect redundant assertions in a mapping, or to test equivalence of candidate mappings. As an example, recall mapping {*m*1, *m*2} from Example 1. Assertions *m*1, *m*2 are each one inferred from mapping {*m*3}, but assertion *m*3 is not inferred from {*m*1, *m*2}.

Definition 3 (*Mapping Inference*). Let *M* be a mapping from schema *S* to schema *T*. Let *F* be an assertion from *S* to *T*. We say *F* is *inferred* from *M* iff $\forall I_S$, I_T instances of schema *S* and *T*, respectively, such that I_S and I_T satisfy the assertions in *M*, then I_S and I_T also satisfy assertion *F*.

As with the previous property, the flat database schema that encodes the mapping scenario is:

$$DB = (PD_S \cup PD_T,\ DR_S \cup DR_T \cup DR_M,\ IC_S \cup IC_T \cup IC_M)$$

In order to reformulate mapping inference in terms of query satisfiability, we must get rid of the universal quantifier that appears in the property's definition. The reason is that by means of query satisfiability we can check whether there exists an instance that satisfies the property encoded by the query, but not whether all instances satisfy that property. We can address this situation by checking the negation of the property instead of checking the property directly; that is, we will check whether there is a pair of schema instances that satisfy the mapping but not the given assertion.

If the assertion to be tested is a query inclusion, i.e., $Q_{source} \subseteq Q_{target}$, then the query to be tested satisfiable on *DB* is defined by a single deductive rule:

$$Q_{notInferred} \leftarrow \neg\text{T-inclusion}(Q_{source}, Q_{target}, \varnothing)$$

If the assertion to be tested is a query equality, i.e., $Q_{source} = Q_{target}$, then the query to be tested satisfiable on *DB* is defined by two deductive rules:

$$Q_{notInferred} \leftarrow \neg\text{T-equality}(Q_{source}, Q_{target}, \varnothing)$$
$$Q_{notInferred} \leftarrow \neg\text{T-equality}(Q_{target}, Q_{source}, \varnothing)$$

Boolean query $Q_{notInferred}$ is satisfiable over *DB* if and only if the given assertion *F* is not inferred from mapping *M*.

Fig. 2 shows an instantiation of the example mapping scenario in Example 1 which satisfies mapping {m_1, m_2} but not assertion m_3, i.e., the instantiation is an example that illustrates m_3 is not inferred from {m_1, m_2}.

7. Related Work

In this section, we compare our approach with the previous works on nested relational mapping validation and on translating nested queries into the flat relational setting.

7.1 Mapping Validation on Nested Scenarios

Previous work on mapping validation on the nested relational setting has mainly focused on instance-based approaches: the Routes approach [7], the Spicy system [6], and the Muse system [1]. These approaches rely on specific source and target instances in order to debug, refine and guide the user through the process of designing a schema mapping, which do not necessarily reflect all potential pitfalls.

The Routes approach requires both a source and a target instance in order to compute the routes. The Spicy system requires a source instance to be used to execute the mappings, and a target instance to compare the mapping results with. The Muse system can generate its own synthetic examples to illustrate the different design alternatives, but even in this case the detection of semantic errors is left to the user, who may miss to detect them.

Routes, Spicy and Muse allow both relational and nested relational schemas with key and foreign key-like constraints—typically formalized by means of tuple-generating dependencies (TGDs) and equality-generating dependencies (EGDs)—, and mappings expressed as source-to-target TGDs [20]. Muse is also able to deal with the nested mapping formalism [15], which allows the nesting of TGDs. Comparing with our setting, the class of disjunctive embedded dependencies (DEDs) with derived relation symbols and arithmetic comparisons that we consider includes that of TGDs and EGDs. That is easy to see since it is well-known that traditional DEDs already subsume both TGDs and EGDs [11]. Similarly, our mapping assertions go beyond TGDs in two ways: (1) they may contain negations and arithmetic comparisons, while TGDs are conjunctive; and (2) they may be bidirectional, i.e., assertions in the form of $Q_A = Q_B$ (which state the equivalence of two queries), while TGDs are known to be equivalent to global-and-local-as-view (GLAV) assertions in the form of $Q_A \subseteq Q_B$ [13].

Outside the nested relational setting, other works have proposed and studied desirable properties for different classes of XML mappings.

In [3], the authors study the *consistency* checking problem for XML mappings that consist of source-to-target implications of tree patterns between DTDs. Such a mapping is consistent if at least one tree that conforms to the source DTD is mapped into a tree that conforms to the target DTD. This work extends the previous work of [4], where mapping consistency is addressed for a simpler class of XML mappings.

The mapping consistency property of [3] is very similar to our notion of mapping satisfiability; the main difference is that we introduce the requirement

that mapping assertions have to be satisfied in a non-trivial way, that is, a source instance should not be mapped into the empty target instance. We introduce this requirement because the class of mapping scenarios we consider—with integrity constraints, negations and arithmetic comparisons—makes likely the existence of contradictions either in the mapping assertions, or between the mapping assertions and the schema constraints, or between the mapping assertions themselves; which may result in mapping assertions that can only be satisfied in a trivial way.

7.2 Translation of Assertions with Nested Queries into Flat Relational

Since our mapping assertions are in the form of query inclusions and query equalities, the problem of translating these assertions into the flat relational setting matches the problem of reducing the containment and equivalence check of nested queries to some other property check over flat relational queries. The works in this latter area that are closer to ours are [18, 12, 8].

In [18], Levy and Suciu address the containment and equivalence of *COQL* queries (*Conjunctive OQL queries*), which are queries that return a nested relation. They encode each COQL query as a set of flat conjunctive queries using indexes. An *indexed query* Q is a query hose head is in the form of $Q(\overline{I}_1; ...; \overline{I}_d; V_1, ..., V_n)$, where $\overline{I}_1, ..., \overline{I}_d$ denote sets of *index variables*, and variables $V_1, ..., V_n$ denote the resulting tuple. Relying on the concept of indexed query, Levy and Suciu define in [18] the property of query simulation, and reduce containment of COQL queries to an exponential number of query simulation conditions between the indexed queries that encode them. Levy and Suciu also define the property of strong simulation [18], and reduce equivalence of COQL queries which cannot construct empty sets to a pair of strong simulation conditions (equivalence of general COQL queries is left open).

In [12], Dong et al. adapt the technique proposed by Levy and Suciu [18] to the problem of checking the containment of conjunctive XQueries. They also encode the nested queries into a set of indexed queries, and also reduce the containment checking to a set of query simulation tests between the indexed queries. Dong et al. also propose some extensions to the query language, such as the use of negation and the use of arithmetic comparisons. They however do not consider both extensions together as we do, and they do not consider the presence of integrity constraints in the schemas.

In [8], DeHaan addresses the problem of checking the equivalence of nested queries under *mixed semantics* (i.e., each collection can be either set, bag or normalized bag). DeHaan proposes a new encoding for the nested queries into flat queries that captures the mixed semantics, and proposes a new property: *encoding equivalence*, to which nested query equivalence under mixed semantics can be reduced to. Notice that this approach is different with respect to ours in the sense that it focus on mixed semantics while we focus on set semantics ([18, 12] focus on set semantics too). We consider set semantics since it makes easier the generalization of our

previous results from the relational setting. DeHaan also proposes some extensions to the query language, but he does not consider the use of negation or arithmetic comparisons.

The main difference of the approach followed by these three works with respect to ours is that we do not intend to translate the mapping assertions into some condition over conjunctive queries. Instead, we propose a translation that takes into account the class of queries and constraints the CQC method is able to deal with, especially the fact that the CQC method allows for the use of negation on derived atoms. We take advantage of this feature and propose a translation that expresses the definition of query inclusion and query equality into first-order logic, and then rewrites it into the syntax required by the CQC method by means of algebraic manipulation. We finally obtain a set of integrity constraints (DEDs) that model the semantics of the mapping assertions and that allows us to encode the mapping when we reformulate mapping validation in terms of query satisfiability.

8. Conclusion

We follow and approach to mapping validation that allows the designer to check whether the mapping satisfies certain desirable properties. We focus in this paper on how to apply this approach to the validation of nested relational mapping scenarios in which mapping assertions are either inclusions or equalities of nested queries. We encode the given nested relational mapping scenario into a single flat database schema. That includes the flattening of the mapped schemas and the mapping's queries, and the encoding of the mapping assertions as integrity constraints. Then, we take advantage from our previous work on validating flat relational mappings [22] and reformulate each desirable property check in terms of a query satisfiability problem over the flat database schema. The idea is that the nested relational mapping will satisfy a certain desirable property if and only if the query that results from the reformulation is satisfiable on the flat database schema. To solve the query satisfiability problem, we apply the CQC method [14], which, to the best of our knowledge, is the only method able to deal with the class of scenarios that we consider here.

Acknowledgments. This work has been partly supported by the Spanish *Ministerio de Ciencia e Innovación* under project TIN2011-24747.

References

1. Alexe, B., Chiticariu, L., Miller, R. J., Tan, W. C.: Muse: Mapping Understanding and deSign by Example. In: Proc. ICDE, 10-19. (2008)
2. Alexe, B., Tan, W. C., Velegrakis, Y.: STBenchmark: towards a benchmark for mapping systems. PVLDB 1(1), 230-244. (2008)

3. Amano, S., Libkin, L., Murlak, F.: XML schema mappings. In: Proc. PODS, 33-42. (2009)
4. Arenas, M., Libkin, L.: XML data exchange: Consistency and query answering. J. ACM 55(2). (2008)
5. Bernstein, P. A., Haas, L. M.: Information integration in the enterprise. Commun. ACM 51(9), 72-79. (2008)
6. Bonifati, A., Mecca, G., Pappalardo, A., Raunich, S., Summa, G.: Schema mapping verification: the spicy way. In: Proc. EDBT, 85-96. (2008)
7. Chiticariu, L., Tan, W. C.: Debugging Schema Mappings with Routes. In: Proc. VLDB, 79-90. (2006)
8. DeHaan, D.: Equivalence of nested queries with mixed semantics. In: Proc. PODS, 207-216. (2009)
9. Deutsch, A., Ludäscher, B., Nash, A.: Rewriting queries using views with access patterns under integrity constraints. Theor. Comput. Sci. 371(3), 200-226. (2007)
10. Deutsch, A., Tannen, V.: Optimization Properties for Classes of Conjunctive Regular Path Queries. In: Proc. DBPL, 21-39. (2001)
11. Deutsch, A., Tannen, V.: XML queries and constraints, containment and reformulation. Theor. Comput. Sci. 336(1), 57-87. (2005)
12. Dong, X., Halevy, A. Y., Tatarinov, I.: Containment of Nested XML Queries. In: Proc. VLDB, 132-143. (2004)
13. Fagin, R., Kolaitis, P. G., Miller, R. J., Popa, L.: Data exchange: semantics and query answering. Theor. Comput. Sci. 336(1), 89-124. (2005)
14. Farré, C., Teniente, E., Urpí, T.: Checking query containment with the CQC method. Data Knowl. Eng. 53(2), 163-223. (2005)
15. Fuxman, A., Hernández, M. A., Ho, H., Miller, R. J., Papotti, P., Popa, L.: Nested Mappings: Schema Mapping Reloaded. In: Proc. VLDB, 67-78. (2006)
16. Halevy, A. Y.: Technical perspective - Schema mappings: rules for mixing data. Commun. ACM 53(1), 100. (2010)
17. Halevy, A. Y., Mumick, I. S., Sagiv, Y., Shmueli, O.: Static analysis in datalog extensions. J. ACM 48(5), 971-1012. (2001)
18. Levy, A. Y., Suciu, D.: Deciding Containment for Queries with Complex Objects. In: Proc. PODS, 20-31. (1997)
19. Madhavan, J., Bernstein, P. A., Domingos, P., Halevy, A. Y.: Representing and Reasoning about Mappings between Domain Models. In: Proc. AAAI/IAAI, 80-86. (2002)
20. Popa, L., Velegrakis, Y., Miller, R. J., Hernández, M. A., Fagin, R.: Translating Web Data. In Proc. VLDB, 598-609. (2002)
21. Queralt, A., Teniente, E.: Decidable Reasoning in UML Schemas with Constraints. In: Proc. CAiSE, 281-295. (2008)
22. Rull, G., Farré, C., Teniente, E., Urpí, T.: Validation of mappings between schemas. Data Knowl. Eng. 66(3), 414-437. (2008)
23. Rull, G., Farré, C., Teniente, E., Urpí, T.: Validation of schema mappings with nested queries. Technical Report ESSI-TR-12-5 http://hdl.handle.net/2117/16746 (2012)
24. Rull, G.: Validation of Mappings between Data Schemas. Ph.D. Thesis. Universitat Politècnica de Catalunya. http://hdl.handle.net/10803/22679. (2011)
25. Yu, C., Jagadish, H. V.: XML schema refinement through redundancy detection and normalization. VLDB J. 17(2), 203-223. (2008)

Guillem Rull is currently postdoc researcher at the Department of Service and Information System Engineering (ESSI) at the Universitat Politècnica de Catalunya – BarcelonaTech. He received his Ph.D. degree from the Technical University of Catalonia in 2011. His current research interests are involved with schema and mapping validation.

Carles Farré is currently associate professor at the Department of Service and Information System Engineering (ESSI) at the Universitat Politècnica de Catalunya – BarcelonaTech. He received his Ph.D. degree from the Technical University of Catalonia in 2003. He worked on deductive databases, query containment and schema validation. His research interests are involved with conceptual modeling and data and service integration.

Ernest Teniente is a full professor at the Department of Service and Information System Engineering at the Universitat Politècnica de Catalunya – BarcelonaTech, where he teaches graduate and undergraduate courses on software engineering and databases. He got his PhD in Computer Science from the same university. He was a visiting researcher at the Politecnico di Milano and at the Universita' di Roma Tre, in Italy. His current research interests are focused on conceptual modeling, automated reasoning on conceptual schemas and data integration. He is author of more than 50 publications in international conferences and journals in the areas of databases and software engineering, and he is regularly invited to serve on the Program Committees of international conferences in these areas.

Toni Urpí is currently associate professor at the Department of Service and Information System Engineering (ESSI) at the Universitat Politècnica de Catalunya – BarcelonaTech. He received his Ph.D. degree from the Technical University of Catalonia in 1993. He worked on deductive databases, database updates, integrity constraint maintenance, query containment, and schema validation. Current research interests are involved with conceptual modelling and data integration.

Received: July 03, 2012; Accepted: October 04, 2012.

DOI: 10.2298-CSIS12010204IJ

Accessibility Algorithm Based on Site Availability to Enhance Replica Selection in a Data Grid Environment

Ayman Jaradat[1], Ahmed Patel[2], M.N. Zakaria[1], and A.H. Muhamad Amina[1]

[1] Faculty of Science & Information Technology,
Department of Computer & Information Sciences
Universiti Teknologi PETRONAS
ayman418@yahoo.com, {nordinzakaria, ananghudaya}@petronas.com.my
[2] School of Computer Science,
Centre of Software Technology and Management (SOFTAM)
Faculty of Information Science and Technology (UKM)
Universiti Kebangsaan Malaysia (UKM)
43600 UKM Bangi, Selangor Darul Ehsan, Malaysia
[2] Visiting Professor
School of Computing and Information Systems
Faculty of Science, Engineering and Computing
Kingston University
Kingston upon Thames KT1 2EE, United Kingdom

Abstract. A data grid functions as a scalable base for grid services to manage data files and their scattered replicas around the world. The principal objective of grid services is to support various data grid applications (jobs) as well as projects. Replica selection is an essential high-level service that selects a Grid location which verifies the shortest response time for the users' jobs among numerous different locations. In the grid environment, estimating response time precisely is not a simple task. Existing replica selection algorithms consume high response time to retrieve replicas because of miss-estimating replicas transfer times. This paper proposes a novel replica selection algorithm that considers site availability in addition to data transfer time. Site availability has not been addressed in previous efforts in the same context this paper does. Site availability is a new factor that can be utilized to estimate response time more accurately. Selecting an unavailable site or selecting a site with insufficient time will likely lead to disconnection. This in turn will require shifting to another site to resume the download or to start the download from scratch depending on the fault tolerance mechanism. Simulation results demonstrate that the performance of the new algorithm is proved to be better than the existing algorithms mentioned in literature.

Keywords: data grid architecture, grid computing, grid component failure, virtual organization, OptorSim.

1. Introduction

In numerous scientific disciplines, terabyte (possibly soon to be petabytes) scale data collections is emerging as critical community resource. The required "data grid" infrastructure needs to support potentially thousands of users. Especially scientists who want to work collaboratively in their field all over the world [1]. Conversely, it is evident that one virtual organization (VO) alone may not be sufficient to manage the massive volume of data produced from experiments and simulations. Contextually, the exponential growth of scientific applications has opened up a new research horizon for computer scientists and researchers. This can produce efficient techniques and algorithms for scientific applications that require access, storing, transferring, analysis and replication of an immense amount of data in geographically distributed locations [2]. Replication and distribution of data among diverse grid sites are needed to address the requirement to increase data reliability and availability. Replicated data lead to the requisite of replica selection, a process which selects one replica location from among many replicas based on their response times. The response time is a critical factor that influences the job turnaround time. In previous studies, data transfer time was utilized to estimate the response time. However, measuring transfer time alone is insufficient. The continuity of service provided by the selected site plays a major role in assuring that the estimated response time will be maintained and not interrupted. This is due to the local policies of the provider that offers services to outsiders for specific hours only. According to the authors of [3], once a user is allowed to gain access to a resource based the access policy, the usage Service Level Agreement (SLA) determines how much of the resources the user is permitted to use.

Just to recap: in the literature, *availability* signifies the production of a number of copies for a single file (resource) in order to make it constantly available [4]. *Availability* in this research is defined as the capability of a given resource to fulfill a given task until it is completed. To distinguish between these two definitions, we use *site availability* **or** *accessibility* to refer to the second definition.

In [5] it is reported that only 65% of users' submitted jobs are executed successfully due to unknown causes of failure. The main causes of failures within grid infrastructures are grid component failures, network failures, information faults, and excessive delays. Grid component failures involve both software and hardware account for 25%-30% of the total failures. However, according to [6] the Open Science Grid (OSG) [7], encountered a 30% job submission failure rate with 90% of them due to disk filling errors, gatekeeper overloading, and network disruptions. Though many enhancements have been done, the grid keeps growing in both size and complication. The total improvements are often not enough: for instance, the LCG grid [8] is still reporting about a 25% error rate [9]. Troubleshooting grid middleware is very challenging due to large number of interconnected components. For example, one action, like reliably transmitting a directory of

files, could result in the coordination of a wide-ranging collection of loosely coupled software tools. Each of them normally generates its own log files in their own log format, semantics, and identifiers. To troubleshoot a problem as it cascades from one component into the next, this information must be combined to form a logically consistent trail of activity.

Causes of failures are mostly vague and request further investigations. Although, we can conjecture that excessive delays and the insufficient time of the resources to complete tasks are among of the reasons[1].Therefore integrating site availability in the replica selection process is necessary to avoid such faults or delays. To the best of our knowledge, none of the researchers have introduced site availability with the same concept that we have specifically detailed in this research. Site availability is defined as: *The relationship between the operating time declared by the service provider to serve certain VOs and the required time to transfer a file from the same provider during the replica selection decision process.*

This study tries to highlight that incorporating site availability as a new intervention for a deliberated estimation of response time enhances the data grid environment. Incorporating site availability as a selection factor in replica selection algorithm provides replication management systems with more guaranteed response time estimation.

2. Related Works

Data replication modeling has received increasing attention especially in the past few years. Replica selection algorithm is one of the major functions of replication management system which determines the best replica location for grid users. Such determination is critical because the resources are limited and users competing for it. Replica selection algorithms are categorized into two groups namely partitioned and greedy. Partitioned algorithms [10-12] are classified into two sets namely 'available' and 'unavailable'. The forecasted server latency is computed for each replica and compared with a pre-calculated threshold value to categorize replicas into 'available' or 'unavailable'. In greedy algorithms [13, 14], the client is assigned to a replica, which is forecasted to provide the best transaction performance. This transaction performance needs to be estimated before selecting the most optimum replica. On the other hand weighted algorithms [14, 15] estimate the proportional rate of assigning a user to a certain replica based on the weight assigned to each of these replicas. The authors of [16] have proposed a variety of replication strategies, which are evaluated on

[1]Each site has its operating hours to serve the others based on its local policy. However accessing sites which are available for shorter time than required will lead to timeout and this obliges to complete or to restart the task the in another site if such mechanism is available. Sometimes also the problems occur and it is very difficult to know or to trace the causes.

hierarchical grid architecture. The proposed replication algorithms are based on the hypothesis that popular files of one location will also be popular at another location. The number of hops for each site that houses the replica is considered. The best replica is the one that requires the minimum number of hops to reach the requesting user. On the other hand, the authors of [17] used the log files of the Grid File Transfer Protocol (GridFTP) only as the tool to predict the replica with the fastest response time. However, in [18] the researchers have proved that GridFTP alone is insufficient for the best prediction. Preferably, a regression technique model should be constructed to forecast the data transformation time from the source to the destination based on the three data points, mainly GridFTP, Network Weather Service (NWS), and I/O Disk. On the other hand, the researchers in [19] have proposed the K-Nearest Neighbor (KNN) rules. This KNN selects the best replica by taking into consideration the history of transferring the preceding replicas which is collected from the logs' files. They also proposed a predictive procedure to estimate transfer time between sites via neural networks.

In [20] the researchers conceived a fuzzy logic technique to evaluate the replication "state" (i.e., negative, normal and/or positive) using the gray prediction model to analyze the factors that affect replica selection but site was not their concern.

Some other works have focused on utilizing parallel techniques to reduce replica transfer time. Their approaches retrieved replicas concurrently from all the available sites [20, 21] that housed that replica. In such approaches, the required file was divided into parts and each part would be retrieved from different servers. The authors of [21] proposed a new grid data-transfer tool (rFTP) that retrieves partial segments of data in parallel.

The authors of [22] devised a PU-DG Optimizer toolbox (also recognized as PU-DG Optibox) that is a package containing some efficient techniques and algorithms. The algorithms are operating as middleware on the top of data grid platforms to optimize file downloads by improving its effectiveness and performance. The toolbox allows the users to select their preferences. It adopts three network factors including bandwidth (B), distance (D), and history record (H). Therefore, the preferences have totally six different options: BDH, BHD, DBH DHB, HBD, and HDB, in which the user can choose one. The toolbox utilizes mathematical formulations for download time. It is transformed into dynamic programming problem, in order to reduce the final time complexity to O(n), where n is the number of candidate replica sites. The toolbox also provides manual and automatic download modes for users, independently whether they are experts or not in computing. It is anticipated that such an approach could decreases the problems that most users could possibly face, of operating and managing files in a data grid environment.

Some recent works [23-26] have addressed the notions of utilizing security to select resources in a grid environment and others have integrated it with replica transfer time to identify the best replica. They defined security in different ways, namely: trust, self-protection, reputation and reliability.

While there have been several works on replica selection, none to the best of our knowledge incorporates the site availability as a factor that influences response time. Moreover, none has considered site availability as a selection factor.

3. System Design

The architecture of the data grid services is divided into two levels as shown in Figure 1 [1]. The upper level includes the high-level services that utilize the low-level or core services. Replica selection optimization technique is high-level service so it invokes a number of core services. Information about an individual resource or set of resources is collected and maintained by a Grid Resource Information Service (GRIS) daemon [27]. GRIS is designed to gather and announce system configuration metadata describing that storage system. For example each storage resource in the Globus data grid [1] incorporates a GRIS to circulate its information. Typically, GRIS informs about attributes like storage capacity, seek times, and description of site-specific policies governing storage system usage. Some attributes are dynamic varying with various frequencies such as total space, the available space, queue waiting time and mount point. Others are static such as disk Transfer Rate.

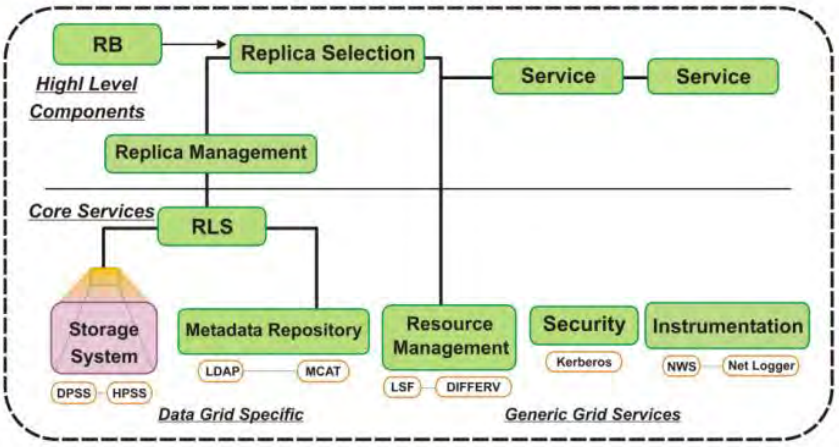


Fig. 1. Major components and structure of data grid architecture. (Adopted from[1])

The new algorithm, as illustrated in Figure 2, commences by receiving the user request via the Grid Resource Broker (RB). RB then retrieves related physical file names and locations from the Replica Location Service (RLS). Subsequently, the algorithm receives information about the sites which hold the replicas and their network status from the GRIS such as: Network

Weather Service (NWS)[2] [28], Meta-computing Directory Service (MDS) [29] and Grid File Transfer Protocol (GridFTP) [29]. Then, the best replica site for the concerned user's job is chosen. In this context, the replica that promises the minimum response time with the least probability of disruption is the best. Hence, the new high-level service replica selection algorithm is an optimization approach. The proposed algorithm is designed to perform caching not replication. Caching [30] occurs on the user side; the user decides which replica is the best and copies the required replica to the local site. On the other hand, replication occurs on the server side; the server that houses the replicas decides which replicas are to be created and where to place them.

The exact sequence of steps in the proposed algorithm is as follows:

- Collects jobs from the Resource Broker.
- Collects replica of physical file names and locations from Replica Location Service.
- Collects sites' operating hours from their log files.
- Collects sites' current criteria values like bandwidth from the information service providers for instance GridFTP, NWS, and MDS.
- Calculates the response time and site availability of each site and rates them by percentage. The site that demonstrates the best *Response Time* (*T*) will be given the value of 100% and the rest of sites will be rated based on their performance in comparison to the site that gets 100%. On the other hand, the rank of site availability 100% will be given to the site or the sites that show sufficient time to complete the transfer even if the dynamic conditions of the network are degraded to some extent. A site is assigned 100% site availability if it shows availability equal to the predicted download time pulse the reserve time required to accommodate any decline in the network. Site availability of the remaining sites is rated based on the predicted download time and how much time is required for the reserve time.
- Selects the best location that houses the required replica for the grid user. The best location is the one that shows minimum transfer time and the least probability of failure to complete the job due to site downtime.

This study focuses on incorporation of site availability as an essential element in the process of locating the best replica. Site availability in this work is defined as the relationship between the required time to download a replica and the remaining time declared by the site that offers this service. The remaining time of any site is the remaining over time to serve the user. The response time is defined as the time elapsed when moving data file from one site to another. The following subsections detail the calculation of site availability, response time, remaining time and the best site selection:

[2] NWS conducts end-to-end network probes (which it uses to measure available network performance) and then applies fast statistical models to probe histories to make performance forecasts

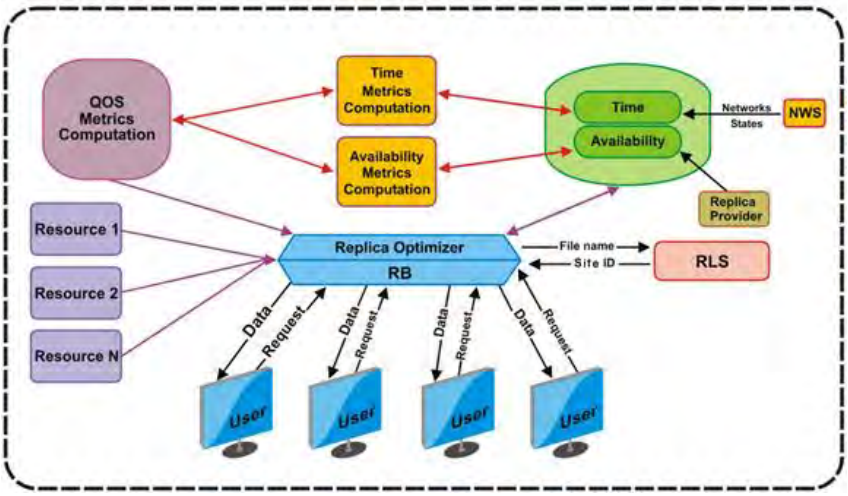


Fig. 2. An overview of the new proposed algorithm

3.1. Calculating Time

Response time is a dynamic value changing as time passes based on the load on the network or the storage devices. However it is anticipated to be steady for a while or change slightly positively or negatively. But since it is difficult to estimate the response time in a dynamic manner, the response time can be estimated at the decision time (NWS applies fast statistical models to probe histories to make performance forecasts). The response time's dynamicity is considered by integrating the new factor site availability (more details about site availability is in subsection B). The response time for a given site *i* is estimated by using the following equations proposed in a recently published work [31]:

$$Ti = T1i + T2i + T3i. \quad (1)$$

$T1$ represents the transfer time, $T2$ represents the storage access latency and $T3$ represents the requested waiting time in the queue. $T1$ represents the data transmission via a wide area network, which depends on the network bandwidth, either a wide area network (WAN) or a local area network (LAN) and the file size which is computed by the following equation [32]:

$$T_{1i} = \frac{FileSize\,(\mathrm{MB})}{Bandwidth\,(MB/SEC)}. \quad (2)$$

In general, the operating systems schedule the disk I/O requests in a manner that improves system performance [33]. The process of scheduling is implemented by maintaining a queue of requests for the storage device. Therefore, the storage speed and the number of requests in the queue play a major role in the average response time experienced by applications. As a result, storage access latency (T_2) is the delayed time of the storage machines to cater the requests and the delayed time depending on the file size and the storage type. Hence, T_2 increased due to larger data files. Moreover, different storage machines have discrepant speeds (data transfer rates) during I/O operations. For example, a tape drive is slower than a disk pool and there are many types of tape drives with different speeds. For instance: the Hewlett-Packard (HP) Storage Works Ultrium 920 Drive speed = 120 MegaBytes per second (*MBps*) while the HP Storage Works Ultrium 448 Drive speed = 24 *MBps* [31]. Storage access latency (T_{2i}) is calculated using the following equation:

$$T_{2i} = \frac{File\,\text{Size (MB)}}{\text{Storage Speed (MB/SEC)}}. \quad (3)$$

Storage machines receive many requests at the same time, but they can only serve one request at a time. This leads to pending the requests on waiting in the queue. Input data transfer must be performed prior to an actual request. Similarly, output data transfer must be completed after an actual write process request. This buffering technique balances required time for requests waiting in the queue and the required time for storage media to serve the request in process [33]. Furthermore, the site will be busy during the period that it transfers any replica from the storage machine to the network. Any new incoming data requests have to wait for the transaction to complete and for the requests that join the queue prior to the underlying request [32]. Consequently, the new request should wait all the earlier requests to be processed in the storage queue. The waiting time is the sum of time from the first request in queue to the last. Each of these times is the storage access latency time T_2. The request waiting time in queue (T_{3i}) is calculated using the following equation:

$$T_{3i} = \sum_{i=1}^{n} T_{2i.}. \quad (4)$$

(*n)* represents the number of requests which are waiting in the queue prior to the underlying request. To make it simple, this work assumes the queuing model is M/M/1/N Poisson arrivals and service. The queuing model represents a single server which has a waiting queue only for N customers (including the one in service). The discipline is the first come, first served (FCFS) [34]. Substituting Equations 2, 3 and 4 in Equation 1 produces:

$$T_i = \left\lceil \frac{File\,\text{Size}}{\text{BW}} \right\rceil + \left\lceil \frac{File\,\text{Size}}{\text{Storage Speed}} \right\rceil + \left\lfloor \sum_{i=1}^{n} T_{2i} \right\rfloor. \quad (5)$$

However, it is worth mentioning that modern storage systems with disks and flash memories allow networking and storage to occur simultaneously. Hence, Equation 5 is modified as follows:

$$T_i = \text{MAX}\left\{ \left\lceil \frac{File\,\text{Size}}{\text{BW}} \right\rceil + \left\lceil \frac{File\,\text{Size}}{\text{Storage Speed}} \right\rceil + \left\lfloor \sum_{i=1}^{n} T_{2i} \right\rfloor \right\}. \quad (5a)$$

Table 1. 10 GB and 100 GB replicas with different metric values for: common storage speed and bandwidth, queue waiting time and remaining time

#	1	2	3	4	5	6	7	8	9	10	11
	File size (GB)	Storage speed (MBps)	Bandwidth (MBps)	Queue waiting time (S)	Estimated transfer time (S)	Time rated Out of 100%	Remaining Operation Time (Rs)	Availability Rated out of 100%	Quality Distance *qd*	Scaled Standard Deviation	Best replica *TA*
1	10	150	45	0	295	36	500	84	46	3.39	49.39
2	10	300	156	150	249	43	300	60	49	1.20	50.20
3	10	600	622	300	333	32	70	10	79	1.56	80.56
4	10	300	156	10	109	100	200	91	6	0.64	6.64
5	10	600	622	1200	1233	8	2500	100	65	6.51	71.51
6	10	150	45	100	395	27	400	50	62	1.63	63.63
7	10	600	622	75	108	100	150	69	21	2.19	23.19
8	10	150	45	100	395	27	600	75	54	3.39	57.39
9	10	300	156	200	299	36	150	25	69	0.78	69.78
10	100	150	45	0	2958	21	2500	42	69	1.48	70.48
11	100	300	156	150	1147	55	2000	87	33	2.26	35.26
12	100	600	622	400	745	86	500	34	47	3.68	50.68
13	100	300	156	0	997	63	2000	100	26	2.62	28.62
14	100	600	622	600	935	67	1000	53	40	0.99	40.99
15	100	150	45	100	3058	20	3700	60	63	2.83	65.83
16	100	600	622	700	1035	61	1500	72	33	0.78	33.78
17	100	150	45	1200	4158	15	8500	100	60	6.01	66.01
18	100	300	156	400	1397	45	1900	68	44	1.63	45.63

Therefore the replica selection algorithms should be aware of the technology utilized in each site in order to estimate its response time accurately. However, the proposed algorithm is not limited to using the

abovementioned data transfer speed models; any other valid model could easily replace the above mentioned models as an alternative solution.

Rating sites based on their response time (T_0i) is denoted by the following equation:

$$T_{0i} = \frac{\min\{_{i=1}^{n} T_i}{T_i} \times 100. \quad (6)$$

For example, as shown in Table 1 which reflects real bandwidth, storage speeds and file sizes, the estimated download time based on Equation 5 from sites 1, 2, 3 and 4 are 295s, 249s, 333s and 109s respectively. Site 4 displays the minimum download time so it is rated as a 100% site, site 2 is rated based on Equation 6, $\frac{109}{295} \times 100 = 36\%$ while site 3 is rated $\frac{109}{249} \times 100 = 43\%$ and site 3 is rated $\frac{109}{333} \times 100 = 32\%$. As a result, all sites are rated based on estimated download time to make the selection decision in the next step feasible and easier. The content of Table 1 will be discussed in detail in the following subsections.

3.2. Calculating Site Availability

Site availability is the relationship between the operating time declared by the service provider to serve certain VOs and the time required to transfer a file from the same provider during the replica selection process. Therefore, site availability (A) is computed as follows:

1. Ascertaining the remaining operation time (or allowed time) in seconds (Rs) from the site.
2. Estimating the required time to transfer the file (Ts).
3. Site availability is calculated by:

$$A = \frac{Rs\,(\mathrm{SEC})}{\mathrm{Ts}\,(\mathrm{SEC}) \times 2\alpha}. \quad (7)$$

The value of α is measured based on the network expected performance and the expected download time as well. The replicas usually are very large in size that is why they require long time to be downloaded. During this time, the network performance is prone to change either negatively or positively. The more stable the network condition is, the smaller value of α is required. For example, if the network performance shows that the real time to transfer a file is two times more than the estimated transfer time Ts, then α should be equal to 2. The value of α can be obtained based on some factors like: place, workdays, holidays, weekends, mornings, evenings, midnights and the comparison of file transfer history and estimated time transfer history. The minimum value of α should not be less than one. This is when the replica

download time estimation is 100% accurate; the value of α is obtained from the history information by comparing the estimated transfer times with the actual transfer times. On the other hand, the maximum value of *A* should not exceed 100% because it is adequate and more than 100% is considered overqualified, which adds no values as demonstrated in Equation 8. In the example below, we assigned the value 1 to α, assuming 100% accuracy in download time estimation. However, based on our approach this number should be multiplied by 2 in order to be more confident that the transfer will commence and terminate from the same site and to avoid any risk of disconnection prior to download completion as shown in Equation 7. Hence, the minimum acceptable value for *A* is 50% but a higher value increases the success rate. On the other hand, estimating α requires more attention, which is outside the scope of this study. We plan to address this estimation issue in future work.

Site availability is rated as follows:

$$A_0 = \begin{cases} 100 & ,\mathrm{Rs} \geq \alpha \mathrm{TS} \\ \dfrac{Rs\,(\mathrm{SEC})}{\mathrm{Ts}(\mathrm{SEC}) \times 2\alpha} \times 100 & ,\ \mathrm{Rs} < \alpha \mathrm{Ts} \end{cases}. \quad (8)$$

For example, using the same data shown in Table 1, the estimated download time based on Equation 5 from sites 1, 2, 3 and 4 are 295s, 249s, 333s and 109s respectively and the remaining operating time for each are 500s, 300s, 70s and 200s respectively. Assuming the value of α is 1, the site availability for site 1 is $\frac{500}{295 \times 2 \times 1} \times 100 = 843\%$, and the site availability for site 2 is $\frac{300}{249 \times 2 \times 1} \times 100 = 60\%$. The rest of the calculations are shown in Table 1.

3.3. Estimating the Best Site

The new approach proposes an imaginary ideal or model value to be 100% Time (*T*) and 100% Site availability (*A*) as shown in Figure 3. The best site is the one with the closest distance (*d*) to the ideal value (*T* in Figure 3). We titled it as the quality distance (*qd*) which is calculated using the following equation:

$$qd = \frac{\sqrt{\left(100 - T_0\right)^2 + \left(100 - A_0\right)^2}}{\sqrt{2}} \quad (9)$$

The distance in Equation 9 is divided by $\sqrt{2}$ to normalize its value to be between 0 and 100. The smaller the *qd* value, the better the site.

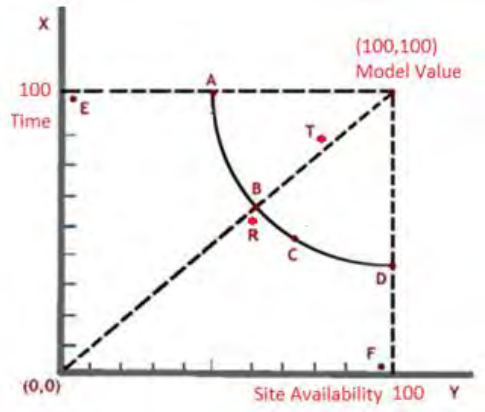


Fig. 3. Visual representation for sites and their model values

As shown in Figure 3, site T is the best site because it is the closest to the model value. If we do not have site T, the algorithm will select site A, B, C or D randomly because they all have the same distance from the Model value. In fact, the best in this scenario is site B because it is composed of two similar or almost similar values. This signifies a balanced solution, which is not extreme for site availability or transfer speed as opposed to site F. Site F displays high site availability but low quality transfer speed, which is still better than site A. Site A, displays high-quality transfer speed and low-quality site availability which could lead to a fault (disconnection). Moreover, it is clear that site R is better than sites A, C and D. To select a balanced solution and to avoid the extreme values as experienced in sites A or D as illustrated in Figure 3, the standard deviation (*sd*) is conceptualized by yielding a balanced optimal composition of time and availability. For example *sd* (70,70) = 0, *sd* (50,50) = 0, *sd* (30,30) = 0 while *sd* (60,40) = 14.14 and *sd* (70,30) = 28.28, and thus, the new equation for finding *qd* is modified to be as follows:

$$mqd = qd + \mathrm{sd}\left(\mathrm{T}_0 + \mathrm{A}_0\right) = \frac{\sqrt{\left(100 - T_0\right)^2 + \left(100 - A_0\right)^2}}{\sqrt{2}} + \mathrm{sd}\left(\mathrm{T}_0 + \mathrm{A}_0\right) \quad \textbf{(10)}$$

Where *sd* increases the value of the quality distance which means degrading *qd*, if the values of its parameters are distant as explained in the previous example.

Conversely, our experiments proved that adopting the standard deviation sometimes has side effects that could divert from the optimal solution. For example, if site X has the combination (63,100) for time and availability, utilizing Equation 10, *mqd* = 52.16 and site Y has the combination (61, 72), *mqd* = 40.78 meaning Y is better than X, even when it is clear that X is better than Y for both parameters, site availability and time. This example proves that the standard deviation has side effects and needs to be utilized wisely. To overcome the problem of standard deviation, it has been scaled down by dividing it into a number β as in Equation 11. The result, site X rating is corrected to be better than Y. The other sites' rates were corrected as well to reflect reality. The last version of *mqd* equation is denoted by:

$$mqd = \frac{\sqrt{\left(100-T_0\right)^2+\left(100-A_0\right)^2}}{\sqrt{2}} + \frac{\mathrm{sd}\left(\mathrm{T}_0+\mathrm{A}_0\right)}{\beta} \qquad (11)$$

Estimating the value β was carried out using a comprehensive search for all possible paired values of availability *A* and time *T* (*A, T*). We created a table containing all the possible values of *A* and *T*. The value 50 was assigned to availability in the first column, which is the minimum applicable value when α =1, the second 51 and so on until the last column was given the value 100. We assigned the first row the value 30 for *T* and the second 31 until the last row was assigned the value 100. Table 2 depicts a summary of the real table. The objective is to find a value for β that satisfies the following conditions:

1- Decreases the value of *mqd* (smaller *mqd*, better performance) while moving in the table from top to bottom. It is logical that the pair (50, 95) is better than (50, 30); certainly if we have both options we will choose the former.
2- Decreases the value of *mqd* while moving from left to right because it is logical that the pair (90, 30) is better than (50, 30).
3- Balances, to some extent, the values of *A* and *T*, for example (50, 50) is better than (90, 30) but (60, 44) is the best because 44 is faster than 30 and 60 is safer than 50.

Different values for β has been tried, from 1 onwards; thus far, the conclusion the value of 10 is the best. For instance, as shown in Table 2, beneath row 8 the value of *mqd* increases while *T* increases which is illogical and contravenes condition 1 as well. On the other hand, if we increase the value of β to be greater than 10, (90, 30) will be better than (50, 50) resulting in an unbalanced combination. In actuality, estimating β requires further researches which will be conducted by the researchers in the future. Therefore, at the moment we leave tuning its value to grid administrators and users' preferences because some users prefer speed over reliability or vice versa or a balance of the two. Our preliminary experiments found that the best value for β is 10 as presented in Tables 1, 2 and 3.

Table 2. All possible paired values of availability *A* and time *T* and various values for *β*

	A	T	qd	mqd β=1	mqd β=10	A	T	qd	mqd β=10	A	T	qd	mqd β=10	A	T	qd	mqd β=10	A	T	qd	mqd β=10
1	50	30	60.83	74.97	62.24	60	30	57.01	59.13	70	30	53.85	56.68	80	30	51.48	55.01	90	30	50.00	54.24
2	50	36	57.43	67.33	58.42	60	36	53.37	55.06	70	36	49.98	52.38	80	36	47.41	50.52	90	36	45.80	49.62
3	50	37	56.87	66.07	57.79	60	37	52.77	54.39	70	37	49.34	51.67	80	37	46.74	49.78	90	37	45.11	48.85
4	50	38	56.32	64.81	57.17	60	38	52.17	53.73	70	38	48.70	50.97	80	38	46.07	49.04	90	38	44.41	48.08
5	50	39	55.77	63.55	56.55	60	39	51.58	53.06	70	39	48.07	50.26	80	39	45.39	48.29	90	39	43.71	47.32
6	50	40	55.23	62.30	55.93	60	40	50.99	52.40	70	40	47.43	49.56	80	40	44.72	47.55	90	40	43.01	46.55
7	50	44	53.08	57.33	53.51	60	44	48.66	49.79	70	44	44.92	46.76	80	44	42.05	44.59	90	44	40.22	43.48
8	50	50	50.00	50.00	50.00	60	50	45.28	45.98	70	50	41.23	42.65	80	50	38.08	40.20	90	50	36.06	38.88
9	50	94	35.61	66.72	38.72	60	94	28.60	31.00	70	94	21.63	23.33	80	94	14.76	15.75	90	94	8.25	8.53
10	50	95	35.53	67.35	38.71	60	95	28.50	30.98	70	95	21.51	23.27	80	95	14.58	15.64	90	95	7.91	8.26

The modified distance *mqd* will be titled as *TA* in this study because it is composed of time and site availability and is given a new metric *TA* instead of meter (cm or km) because we are not measuring a normal distance. *TA* is derived from Time and Site availability where the site with the smallest *TA* is the best, since it is the closest to the imaginary ideal value. Table 3 is a mathematical example of our approach where column 1 represents the value of site availability; column 2, the estimated download time; column 3, the distance from the model value; column 4, the standard deviation of the two values for each site (estimated download time and site availability) divided by 10 and column 5, the total of columns 3 and 4. Again, as shown in Table 3, *qd* is the lowest in row 3, with the values 56, 90 *TA* for site availability and time respectively. However, it is clear that a value of 56 for site availability is very dangerous and thus prone to fault. As a result, this is not the best combination, even when the value of time is the highest. Therefore, standard deviation corrects the selection as can be seen in row 1, which shows the values site availability and time values of 68 each, as the best selection and row 2, as the second choice if row 1 is not available. On the other hand, the new algorithm excludes from the selection any site with site availability less than 50. For instance, referring to Table 3, if sites 1 to 5 do not exist and the competition is only between sites 6 and 7, and both of them have the same *TA* value, the winner is site 6 because the site availability for site 7 is less Than 50% which is for sure not enough.

Table 3, Example of applying the proposed algorithm

	A_0	T_0	*qd*	*sd*/10	*qd*+(*sd* /10) (*TA*)
1	68	68	32.00	0.00	32.00
2	65	70	32.60	0.35	32.95
3	56	90	31.91	2.40	34.31
4	60	79	31.95	1.34	33.29
5	50	51	49.50	0.07	49.57
6	60	42	49.82	1.27	51.09
7	42	60	49.82	1.27	51.09

The pseudo code below emphasizes the detailed algorithm:

```
1. get R ( list of physical file names and locations  for
the required replica) from RLS
2. get Rs for each replica from the data grid's log file
3. estimate β
4. i=1
5. while R not empty
     5.1 calculate T0, A0

     5.2 calculate  mqd (i) = sqrt((100−T0)^2 + (100−A0)^2)/sqrt(2) + sd(T0 + A0)/β

     5.3 i = i+1
6. best = mqd(1)

7. j=2
8. While j <= i
    8.1 if  mqd(j) < best & A0(j)  > 50

        8.1.1 best = mqd(j)

9. halt
```

4. Performance Evaluation

To assess the impacts of the new replica selection algorithm, a simulation tool was used to conclude the performance. The researchers, thereby, conducted a comprehensive search on distributed and parallel systems, in particular, simulators that merit grid features [35] for example: MicroGrid, GridSim, SimGrid, OptorSim, Monarc, ChicSim and Bricks. However, OptorSim was found to be the most suitable given that it simulates the replica selection and the data replication strategies [36, 37]. The designer of Optorsim, Figure 4, states that it was developed to "model the interaction of the individual grid components of a running data grid as realistically as possible" [37]. Accordingly, OptorSim has been chosen because it is the most realistic test bed. However has been modified to fit the current research. Figure 4, presents OptorSim architecture.

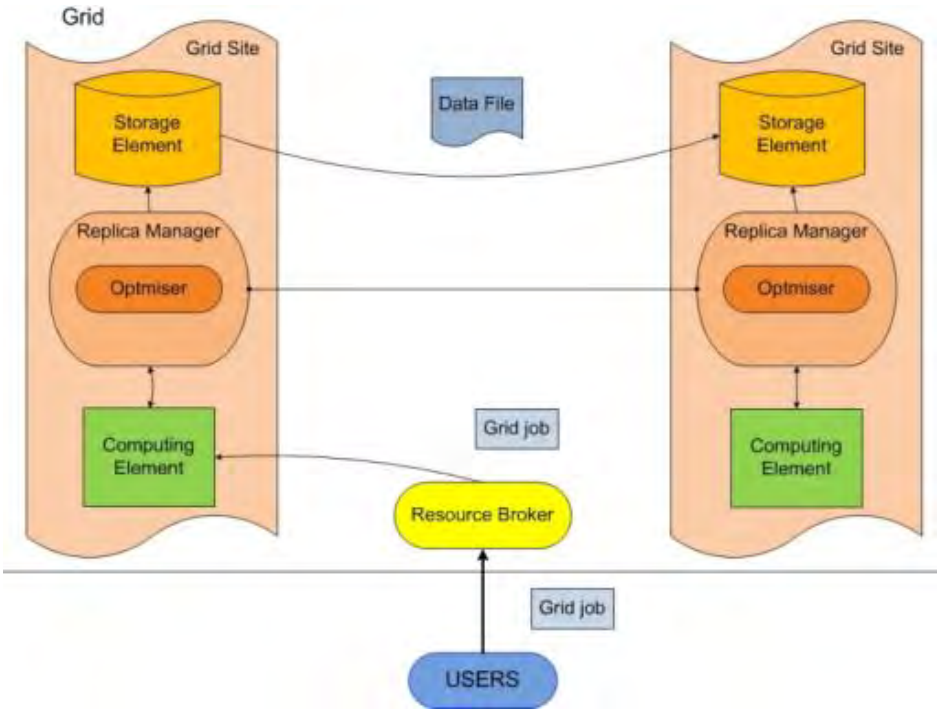


Fig. **4**. OptorSim architecture

5. Simulation Setup

OptorSim is designed as an evaluation tool to test the performance of different job scheduling and replica optimization strategies (a job is usually specified as a set of data files that require analysis). It has a massive number of elements to accomplish in a realistic environment. It contains Computing Elements (CEs) to which the jobs are passed; storage elements (SEs) as a place to keep data; and network elements to connect the grid sites. Like the real grid, bandwidth between sites is integrated in the simulation as well as other network status elements. The remaining two elements are the resource brokers, which submit jobs to grid sites based on scheduling algorithms and the Replication Manager (RM) that plays a role in replication optimization strategies. The OptorSim structure adapts European data grid (EU data grid) topology and configuration. The grid topology as an input to OptorSim consists of 20 sites in the USA and Europe that were utilized during a data production form (CMS test bed) for major LHC experiments [37] as shown in Figure 5 and the other input simulates grid jobs and data file configurations. The European Organization for Nuclear Research (CERN) and Fermi National Accelerator Laboratory (FNAL) are producing the original files and

storing them locally with a storage capacity of 100 GB each and other sites which have at least one CE and a storage capacity of 50 GB each. The order in which a job requests files is determined by the *Access Pattern* used. Some different access patterns have been selected for the simulation. Such as sequential (all files are requested in a predetermined order), Gaussian random walk [37] (successive files are selected from a Gaussian distribution centered on the previous file) and Zipf. A Zipf-like distribution can be regarded as a special kind of exponential distribution allowing the simulation of several types of grid job. Additional essential feature is background network traffic, which can fluctuate variably over time. Any replica selection algorithm has to be flexible enough as to adapt to the constantly fluctuating environment, obtaining the best performance for its users.

The default settings of OptorSim were utilized. They were copied from the EU data grid parameters. The bandwidth between the two sites is marked in Figure 5. In addition, the default OptorSim system workloads' values and parameters' values were utilized as shown in Table 4 (The detailed parameters' values of each site are included in the example folder within OptorSim package. These values represent the real values of the EU data grid).

There are several configuration files used to control various inputs to OptorSim. The grid configuration file describes the grid topology and the content of each site. That is the resources available and the network connections to other sites. The job configuration file contains information on the simulated files, jobs and the site policies for each site (the list of files each site will accept). The simulation parameters file contains various simulation parameters which the user can modify. If the user wishes to simulate background network traffic, a bandwidth configuration file is needed along with several data files to describe the simulated traffic. The simulation accomplished on an Hp desktop with 2.8 G CPU and 2 G RAM. Since OptorSim does not consider site availability, it was amended by assigning service hours to each site ranged from 1second to 24 hours (sites available for less than 1 second are not declared by replica catalog). Thereafter, if the simulator faces a selected replica from a site with insufficient operating time, it will then increase the replica transfer time based on the expected delay. This is done by adding the reconnection setup time (10s) and half of the time consumed to transfer the replica before disconnection because fault tolerance techniques may require resuming or restarting from the beginning. In the simulation the average fault cost is calculated as follow:

$$Fault\ Cost = Rr+Trls+Rd+Cs+Ror \quad (\mathbf{12})$$

Rr: Required time to recognize that there is a fault
Trls: Time to inquire and get the response from RLS
Rd: Replica selection decision time
Cs: Connection setup time
Ror: Resume or restart from scratch time, which is based on fault tolerance technique

In the simulation, *Rr* and *Trls* are set to 2s each, *Rd* 1s and *Cs* set to 5s each. The total is 10s, which is not that critical for usually huge replicas but in contrast, *Ror* has a significant impact especially if the fault tolerance technique requires restarting from scratch. Fault tolerance techniques have an important impact to the replica selection process, which will be addressed in our future work.

Table 4, Workload and system parameter values

Description	Value
Number of files	200
File size	1 G
Storage available at an SE	30 G-100000 G
Number of files accessed by a job	3-20
α	1
β	10

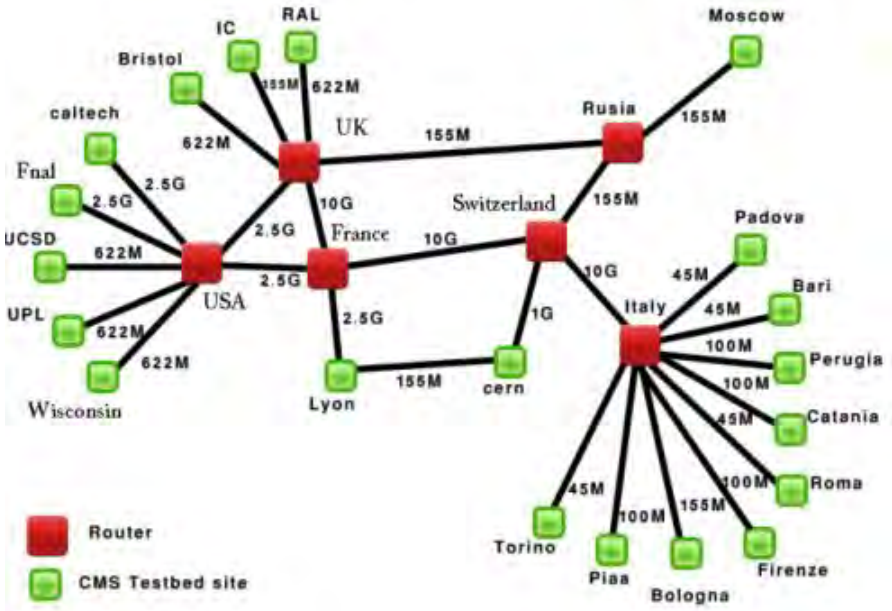


Fig. **5**. Grid topology for CMS test bed

6. Performance Metrics & Cost

In a grid environment, users normally send their jobs to the RB, which locates the best site to carry out the jobs. The executed jobs commonly require some

data files; the optimizer locates the best locations of the required files. However, each site services the users based on their local policy, which allows the users to be served for a specific number of hours per day or night, or even possibly, only on weekends. Hence, selecting the site at an improper time could lead to disconnection. Depending on the fault tolerance approach, the job could be resumed by another site (which may also be prone to disconnection if site availability is not considered or it may be required to restart the entire process from scratch. Therefore, the job's time requirement will increase. The job's time requirement begins from the time the RB transmits the job until the time that the job has completed its execution. This time is called the job turnaround time and includes the response time. The best replica selection according to the new algorithm decreases the response time and consequently decreases the job turnaround time. Therefore, the *Average Job Turnaround Time (AJTT)* is suitable for a performance metric that evaluates our overall algorithm performance and can be measured by using the following equation:

$$AJTT = \frac{\left(\sum_{i=1}^{n} T_{out} - T_{in} \right)}{n} \quad (\mathbf{13})$$

T_{in} represents the time the job is received by the algorithm to begin execution, T_{out} represents the time the job has completed the execution, and n represents the total number of jobs processed through the system. On the other hand, the new algorithm considers two factors to select the best replica. The first is time expenditure and the second is site availability. Therefore, a new quality of service (QoS) value composed of the two factors (*Time* and *Availability*) has emerged and titled *TA*. The lowest value of *TA* means the best quality. Table 1, illustrates scenarios for 10 GB and 100 GB replicas with different metric values for: storage speed, bandwidth, queue waiting time and time remaining. Column 6 shows that the time metrics combinations for the best sites are located in rows 4 and 7 for the 10 GB replica. Row 4 is the best due to high site availability and less disconnection risk. On the other hand, for the file size of 100 GB, the candidate site shown in row 12 reveals the best transfer time of 735s but was discarded because it is available only for 500s, which is not sufficient and a certain error will occur. The optimizer selected the site presented in row 13, which shows 28.62 *TA*. Even the site presented in row number 14 shows a 62s better transfer time. This decision is due to the anticipated high risk from site 14. It is difficult to estimate the cost of our new approach. In the aforementioned example, it was 62s but different situations have different costs but nonetheless, it is worth it for a reliable transfer.

7. Results and Discussion

OptorSim is equipped with different built-in replication strategies (i.e., Least Recently Used (LRU), which always replicates and deletes the least recently used file, Least Frequently Used (LFU), which always replicates and deletes the least frequently used file and the Economic Model-Binomial (EB), which replicates, if it is economically advantageous, using a binomial prediction function for values). However, within these replication strategies only one built-in replica selection algorithm is applied. It selects the best replica locations that show the least transfer time [30-31].

The simulations have been performed to calculate *AJTT* as the average of the total time required for all jobs, measured in seconds. The simulation commenced by investigating the best value for β. Several values were tested for β starting from 1 until 18. On the other hand, due to the fact that there is a strong relationship between α and β, the abovementioned tests were performed utilizing different values for α under LFU replication strategy. Table 5 depicts the results of these experiments wherein the best value of β is 10 when α =1 or 1.5, and the best value of β is 9 when α =2.

Table 5, Average jobs' time in seconds for 500 jobs with different values of α and β

β	*AJTT* when α =1	*AJTT* when α =1.5	*AJTT* when α =2
1	698314.10	1313303.40	1525233.60
2	678414.25	1046605.20	1370006.00
3	650086.10	1023575.94	1335122.10
4	716841.20	1159565.00	1324494.40
5	732999.25	918903.44	1315733.10
6	635794.00	1093129.10	1276658.80
7	680829.75	1418769.50	1376628.80
8	698921.40	978713.56	1353125.50
9	685447.56	1220957.00	1225239.60
10	594141.75	893544.75	1176878.20
11	979156.90	836304.30	1323419.00
12	753310.75	993881.50	1473392.90
13	743662.50	1106562.00	1240304.00
14	634496.50	937375.10	1514095.50
15	734516.50	1029952.30	1438059.80
16	665705.60	1133184.00	1618809.80
17	643339.60	1061195.90	1245205.20
18	801867.10	1104780.40	1318509.40

Moreover, to compare the performance when the systems allows storage and networking to occur simultaneously (Equation 5a), and when it does not allow that (Equation 5), the simulator was operated using both scenarios. The results are illustrated in Table 6. It is clear that overlapping reduces *AJTT*

which reflects better performance. Because the scope of this study is only site availability, and it is anticipated that grid systems are still using legacy storage systems, the remaining experiments were carried out based on Equation 5.

Table 6. Average jobs' times in seconds for 500 jobs based on Equations 5 and 5a

Test #	LUR		LFU		Economic	
	Eq 5a	Eq 5	Eq 5a	Eq 5	Eq 5a	Eq 5
1	12756230	11839082	8973130	11083612	8856054	8628327
2	9173741	8574796	11902111	12664846	9315399	9670981
3	9474930	9842670	10641312	10206533	8758650	9393326
4	8839858	12177088	8817204	12174020	7927166	10171052
5	10249128	10639864	9871939	11989452	9487867	10921624
AJTT	10098777	10614700	10041139	11623692	8869027	9757062
Efficiency based on Eq 5a	4.86%		13.61%		9.10%	

Table 7. Average jobs' times in seconds for 100 jobs when availability is always 100%

Test #	LUR		LFU		Economic	
	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm
1	286111	231099	278278	230458	1060176	906395
2	275035	242358	271168	220449	1085247	1035999
3	284864	222627	255691	223970	913550	904781
4	238518	232944	288959	216565	1005183	954523
5	256388	224360	242888	224768	977966	1062441
AJTT	268183	230678	267397	223242	1008424	972828
Efficiency of the proposed Algorithm	13.99 %		16.51 %		3.53 %	

To verify that the only difference between the proposed algorithm and the built-in in OptorSim is site availability, both algorithms were run with site availability always set to 100%. The expectation was that similar performance would be achieved from both because response time is the only selection factor in the built-in OptorSim and should be in the proposed algorithm when site availability is 100%. However, the simulation results in Table 7 below were surprising. They show that the proposed algorithm is less efficient in all

three of the replication strategies. The justification for that is the number of jobs in this experiment is 100. Each of them is accompanied by 10 to 100 replicas, which means on average around 5500 replicas (decisions). Therefore, there will certainly be some overhead.

Table 8 (a). Average jobs' times in seconds for 100 jobs

Test #	LUR		LFU		Economic	
	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm
1	704189	1278467	628694	1544662	933333	932571
2	582321	1090155	747886	1013950	909764	855317
3	582266	1280359	579806	1243939	836163	946263
4	720064	1105045	706270	1248695	855435	956849
5	650280	1041018	648674	1161950	934881	964598
AJTT	647824	1159009	662266	1242639	893915	931120
Efficiency of the proposed Algorithm	44.11 %		46.70 %		4.00 %	

Based on the abovementioned experiments, the remaining simulation experiments were carried out by setting the value of β to 10. In view of the fact that the number of jobs influenced data transfer time, we evaluated our algorithm's performance in three different scenarios by varying the number of jobs each time. In the first, second and third scenarios, the number of jobs were 100, 500 and 1000 respectively.

We executed the simulation 5 times for each scenario along with a predetermined site operating time scenario, utilizing both our algorithm and the built-in extended algorithm in OptorSim. We did our experiments using three different OptorSim built-in replication strategies, namely, LRU, LFU and EB. Our new replica selection algorithm was tested by performing several executions on the same replicas with a different number of jobs. The results of the simulation demonstrated that the *AJTT* in the new algorithm was less than the *AJTT* of the OptorSim built-in replica selection algorithm for all scenarios and under different replication strategies as shown in Tables 8 (a, b, c), which signified that the proposed algorithm outperformed the previous algorithms.

Table 8 (b). Average jobs' times in seconds for 500 jobs

Test #	LUR		LFU		Economic	
	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm
1	12108976	17167802	9644758	19077332	9065388	8516753
2	11204400	15578618	11485171	12449995	9751129	8698534
3	8571915	17896652	9990595	17365892	8654000	9336498
4	12741477	14618266	11045485	16848332	8335365	10864292
5	9886645	14733334	10801072	13096076	9451450	9190288
AJTT	10902682	15998934	10593416	15767525	9051466	9321273
Efficiency of the proposed Algorithm	31.85%		32.81%		2.89%	

Table 8 (c). Average jobs' times in seconds for 1000 jobs

Test #	LUR		LFU		Economic	
	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm	Proposed Algorithm	OptorSim Built-in Algorithm
1	44425416	59635972	41978100	51684568	30145046	291046946
2	41558504	72720032	44334344	73405680	24623898	24063774
3	41331136	64606356	41013108	67973424	26333746	27421194
4	45374036	60879928	42693512	58488172	28583260	26592294
5	45420436	83777552	42533128	75843184	28765640	26274540
AJTT	43621905	68323968	42510438	65479005	27690318	79079749
Efficiency of the proposed Algorithm	36.15 %		35.08 %		64.98 %	

Figure 6 (a, b, c) depicts the average jobs' total time for the proposed algorithm and the OptorSim built-in algorithm under the replication strategies LRU, LFU and EB where the number of jobs was 100, 500 and 1000 respectively. It is clear that when we increase the number of jobs *AJTT* will be increased regardless of the algorithm or the strategy utilized. However, the increment will be more if site availability is not implemented in the algorithm; this is because the probability of selecting unavailable sites, or sites available for an insufficient amount of time, increases.

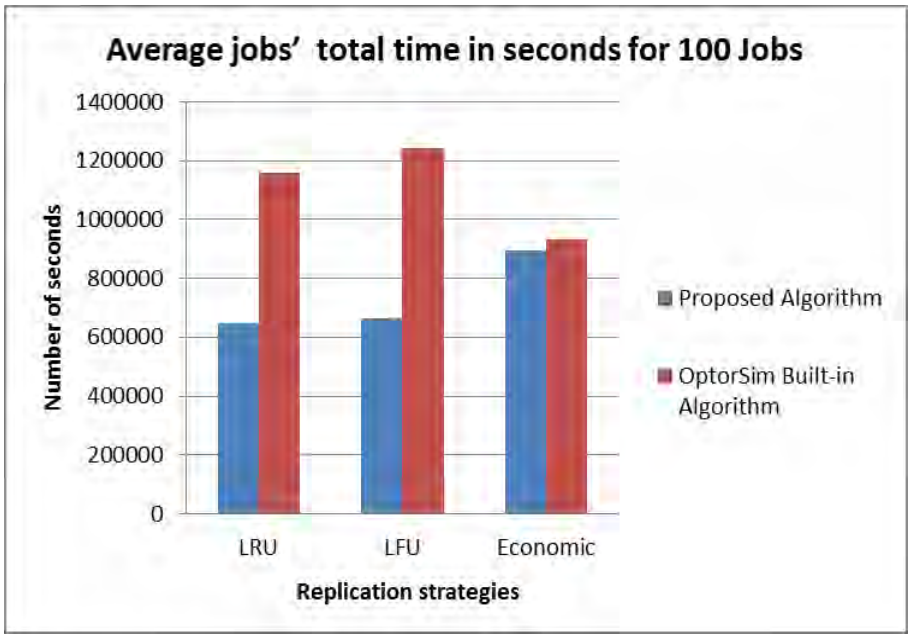


Fig. 6 (a). Average jobs' total time in seconds for 100 jobs

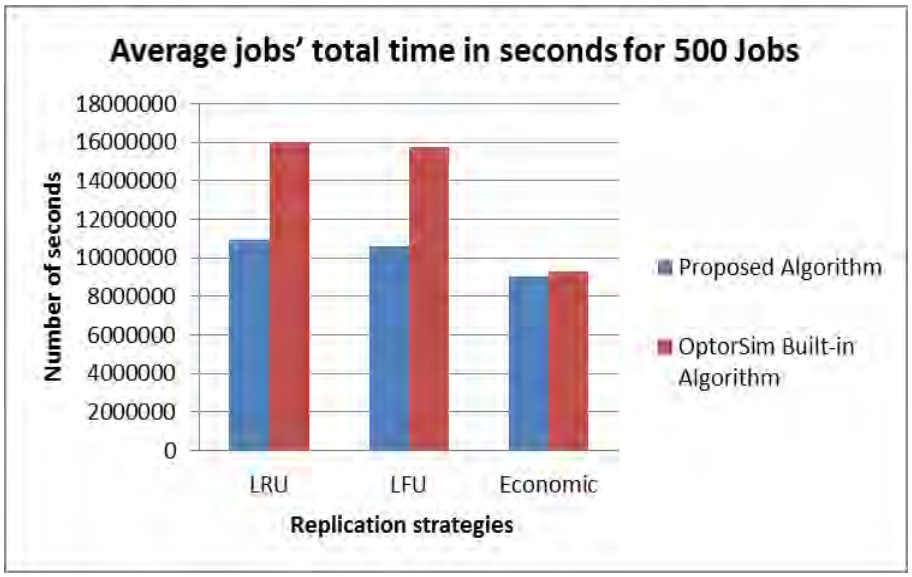


Fig. 6 (b). Average jobs' total time in seconds for 500 jobs

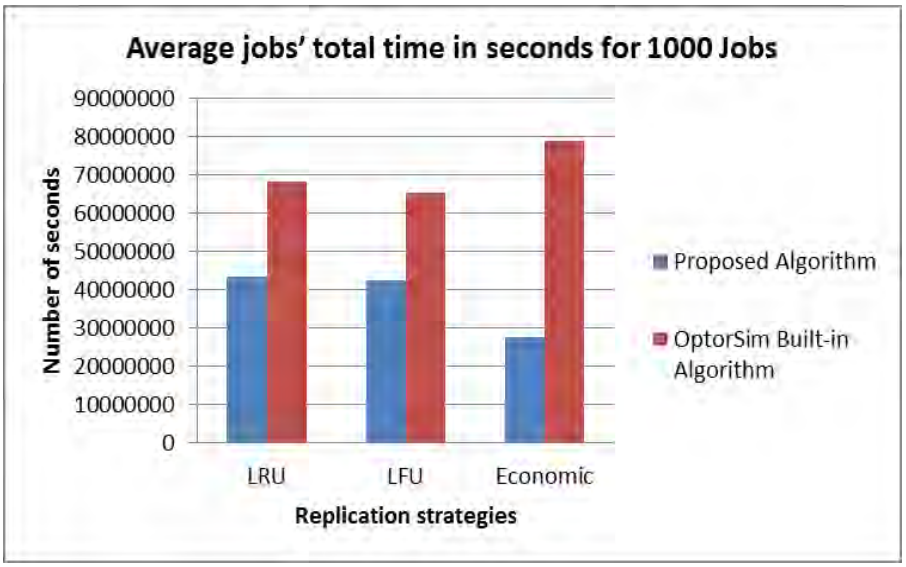


Fig. 6 (c). Average jobs' total time in seconds for 1000 jobs

In real life scenarios, replica selection based only on response time could perform better than the proposed algorithm if the selected sites display insufficient availability but still succeed to deliver the replicas without any disconnection, or if all the sites are available 24 hours per day

8. Conclusion

In this paper, we have introduced a new replica selection algorithm in the data grid environment. The algorithm engaged a new QoS criterion namely site availability in the replica selection process. We defined this novel QoS criterion, demonstrated its importance and integrated it into a replica selection optimizer. A grid simulator (i.e. OptorSim) was utilized to evaluate the algorithm. The simulation experiments were setup by expanding some modules in OptorSim. The strengths of the algorithm had been investigated and the results of our experiments were presented. The simulation results demonstrated that the new algorithm enhanced the performance of the grid environment and thus, decreased the job's average total time. A new network performance parameter α was proposed and its value will be addressed in our future work. Also, the impact of fault tolerance techniques against the download time was highlighted and would be utilized in the replica selection process in our future work.

References

1. S. Vazhkudai, S. Tuecke, and I. Foster, "Replica selection in the globus data grid," in Cluster Computing and the Grid, Brisbane, Qld. , Australia 2001, pp. 106-113.
2. A. Chervenak, E. Deelman, I. Foster, L. Guy, W. Hoschek, A. Iamnitchi, C. Kesselman, P. Kunszt, M. Ripeanu, and B. Schwartzkopf, "Giggle: a framework for constructing scalable replica location services," in 2002 ACM/IEEE conference on Supercomputing, Baltimore, Maryland 2002, pp. 1-17.
3 C. Dumitrescu and I. Foster, "GRUBER: A Grid resource usage SLA broker," Euro-Par 2005 Parallel Processing, pp. 644-644, 2005.
4. M. Lei, S. V. Vrbsky, and X. Hong, "An on-line replication strategy to increase availability in Data Grids," Future Generation Computer Systems, vol. 24, pp. 85-98, 2008.
5. D. Zeinalipour-Yazti and N. Kyriacos, "Managing Failures in a Grid System using FailRank," Department of Computer Science,University of Cyprus2006.
6. I. Foster, J. Gieraltowski, S. Gose, N. Maltsev, E. May, A. Rodriguez, D. Sulakhe, A. Vaniachine, J. Shank, and S. Youssef, "The Grid2003 production Grid: Principles and practice," in 13th IEEE International Symposium on High performance Distributed Computing, 2004, Honolulu, Hawaii, 2004, pp. 236-245.
7. (2011, 6-11). Open Science Grid Consortium. Available: http://www.opensciencegrid.org
8. (2011, 28/10). LCG Grid. Available: http://www.gridpp.ac.uk.
9. M. Aggarwal, D. Colling, B. McEvoy, G. Moont, and O. Aa v. d., "A Statistical Analysis of Job Performance within LCG Grid," presented at the CHEP06, Mumbai, India, 2006.
10. S. Lewontin and E. Martin, "Client side load balancing for the web," in 6th International World Wide Web Conference, Santa Clara, California, 1997, pp. 7-11.
11. Z. M. Fei, S. Bhattacharjee, E. W. Zegura, and M. H. Ammar, "A novel server selection technique for improving the response time of a replicated service," in Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies., San Francisco, CA , USA 1998, pp. 783-791 vol. 2.
12. L. Zuo, S. H. Liu, J. Wei, Y. L. Feng, and G. C. Fan, "Adaptive component replica selection model and algorithms," Ruan Jian Xue Bao(Journal of Software), vol. 19, pp. 1212-1223, 2008.
13. R. Vingralek, Y. Breitbart, M. Sayal, and P. Scheuermann, "Web++: A system for fast and reliable web service," in USENIX Annual Technical Conference, USA, 1999, pp. 13-13.
14. M. Sayal, Y. Breitbart, P. Scheuermann, and R. Vingralek, "Selection algorithms for replicated web servers," ACM SIGMETRICS Performance Evaluation Review, vol. 26, pp. 44-50, 1998.
15. C. Tan and K. Mills, "Performance characterization of decentralized algorithms for replica selection in distributed object systems," in the 5th international workshop on Software and performance, New York, NY, USA 2005, pp. 257-262.
16 R. Kavitha and I. Foster, "Design and evaluation of replication strategies for a high performance data grid," in International Conference on Computing in High Energy and Nuclear Physics, Beijing, China 2001.
17. Y. Zhao and Y. Hu, "GRESS–a grid replica selection service," in 16th International Conference on Parallel and Distributed Computing Systems, Reno, Nevada,USA, 2003.

18. S. Vazhkudai and J. M. Schopf, "Using regression techniques to predict large data transfers," International Journal of High Performance Computing Applications, vol. 17, p. 249, 2003.
19. R. M. Rahman, R. Alhajj, and K. Barker, "Replica selection strategies in data grid," Journal of Parallel and Distributed Computing, vol. 68, pp. 1561-1574, 2008.
20. C. ze Wu, K. gui Wu, M. Chen, and C. X. Ye, "Dynamic Replica selection services based on state evaluation strategy," in Fourth ChinaGrid Annual Conference, 2009, Yantai, Shandong 2009, pp. 116-119.
21. J. Feng and M. Humphrey, "Eliminating replica selection-using multiple replicas to accelerate data transfer on grids," 2004, pp. 356-366.
22. K. C. Li, H. H. Wang, K. Y. Cheng, and T. Y. Wu, "Strategies Toward Optimal Access to File Replicas in Data Grid Environments," Journal of Information Science and Engineering, vol. 25, pp. 747-762, 2009.
23. V. Vijayakumar and R. S. D. W. Banu, "Security for resource selection in grid computing based on trust and reputation responsiveness," International Journal of Computer Science and Network Security, vol. 8, pp. 107-115, 2008.
24. G. Kavitha and V. Sankaranarayanan, "Secure Resource Selection in Computational Grid Based on Quantitative Execution Trust," World Academy of Science, Engineering and Technology, vol. 72, pp. 149-155, 2010.
25. B. Zhang, Y. Xiang, and Q. Xu, "Trust and Reputation Based Model Selection Mechanism for Decision-Making," in Second International Conference on Networks Security Wireless Communications and Trusted Computing, 2010, Wuhan, Hubei 2010, pp. 14-17.
26. S. Naseera, T. Vivekanandan, and K. Madhu Murthy, "Data Replication Using Experience Based Trust in a Data Grid Environment," Distributed Computing and Internet Technology, vol. 1, pp. 39-50, 2009.
27. D. H. Kim and K. W. Kang, "Design and implementation of integrated information system for monitoring resources in grid computing," in 10th International Conference on Computer Supported Cooperative Work in Design Nanjing, 2006, pp. 1-6.
28. R. Wolski, "Dynamically forecasting network performance using the network weather service," Cluster Computing, vol. 1, pp. 119-132, 1998.
29. S. Fitzgerald, I. Foster, C. Kesselman, G. Von Laszewski, W. Smith, and S. Tuecke, "A directory service for configuring high-performance distributed computations," 1997, pp. 365-375.
30. K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid," Grid Computing—GRID 2001, pp. 75-86, 2001.
31. H. H. E. AL-Mistarihi and C. H. Yong, "Response Time Optimization for Replica Selection Service in Data Grids," Journal of Computer Science, vol. 4, pp. 487-493, 2008.
32. K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid," in the Second International Workshop on Grid Computing Denver,CO,2001, 2001, pp. 75-86.
33. S. Aberham, P. Baer, and G. Greg, Operating System Concepts Seventh ed. vol. 5. New York, NY, USA.: Wiley, 1973.
34. S. M. Ross, Introduction to probability models, 6th ed.: Academic Pr, 1997.
35. A. Sulistio, C. S. Yeo, and R. Buyya, "A taxonomy of computer - based simulations and its mapping to parallel and distributed systems simulation tools," Software: Practice and Experience, vol. 34, pp. 653-673, 2004.

36. W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini, "Evaluation of an economy-based file replication strategy for a data grid," in 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid, 2003, Tokyo, Japan, 2003, pp. 661-668.
37. W. H. Bell, D. G. Cameron, A. P. Millar, L. Capozza, K. Stockinger, and F. Zini, "Optorsim: A grid simulator for studying dynamic data replication strategies," International Journal of High Performance Computing Applications, vol. 17, pp. 403-416, 2003.

Ayman Jaradat has obtained his MSc from Universiti Sains Malaysia in 2007 and BSc, Yarmouk University, Jordan in 1989. He is specialized in Computer Science more specifically in distributed systems. His research interest includes grid computing which focuses on data grids, genetic algorithm, distributed algorithms and applications. Jaradat is currently pursuing his PhD at Universiti Teknologi PETRONAS, Malaysia.

Ahmed Patel received his MSc. and PhD. degrees in Computer Science from Trinity College Dublin (TCD), Ireland. He is a Professor in Computer Science at Universiti Kebangsaan Malaysia. His research interests is in Cloud & Grid Computing, Smart Grid, Cyber Security & Digital Forensics. He has published well over 220 technical and scientific papers and co-authored 2 books on computer network security and 1 book on group communications, and co-edited a book distributed search systems for the Internet.

Nordin Zakaria has obtained his PhD from Universiti Sains Malaysia in 2007, MSc from Universiti Malaya in 1999, and BSc from Universiti Putra Malaysia in 1996. Zakaria is specialized in Computer Science and his research interest includes high-performance computing, genetic algorithm, distributed algorithms and applications. Zakaria was assigned to establish and lead the High-Performance Computing Service Center at Universiti Teknologi PETRONAS.

Anang Hudaya Muhamad Amin is a senior lecturer in the Department of Computer & Information Sciences, Universiti Teknologi PETRONAS (UTP), Malaysia. He received a BTech (Hons.) in Information Technology from UTP, Malaysia, and Master of Network Computing and PhD. from Monash University, Australia. His research interests include artificial intelligence with specialization in distributed pattern recognition and bio-inspired computational intelligence, wireless sensor networks and distributed computing.

Received: January 02, 2012; Accepted: October 04, 2012.

DOI: 10.2298/CSIS110927038J

Ant Colony Optimization Algorithm with Pheromone Correction Strategy for the Minimum Connected Dominating Set Problem

Raka Jovanovic[1] and Milan Tuba[2]

[1] Texas AM University at Qatar
PO Box 23874, Doha, Qatar
rakabog@yahoo.com
[2] Megatrend University Belgrade, Faculty of Computer Science
Bulevar umetnosti 29, N. Belgrade, Serbia
tuba@ieee.org

Abstract. In this paper an ant colony optimization (ACO) algorithm for the minimum connected dominating set problem (MCDSP) is presented. The MCDSP become increasingly important in recent years due to its applicability to the mobile ad hoc networks (MANETs) and sensor grids. We have implemented a one-step ACO algorithm based on a known simple greedy algorithm that has a significant drawback of being easily trapped in local optima. We have shown that by adding a pheromone correction strategy and dedicating special attention to the initial condition of the ACO algorithm this negative effect can be avoided. Using this approach it is possible to achieve good results without using the complex two-step ACO algorithm previously developed. We have tested our method on standard benchmark data and shown that it is competitive to the existing algorithms.

Keywords: Ant colony optimization (ACO), Minimum connected dominating set problem, Swarm intelligence, Optimization metaheuristics

1. Introduction

A dominating set for a graph $G(V, E)$ is a subset of vertexes $D \subseteq V$ that has a property that every vertex in G either belongs to D or is adjacent to a vertex in D. Finding the dominating set with the smallest possible cardinality among all dominating sets for a graph is one of the standard NP-complete problems. A very important variation of the minimum dominating set problem is its connected version. We call a dominating set connected if it has the property that any node $n \in D$ can reach any other node $m \in D$ by a path that stays entirely within D. That is, D induces a connected subgraph of G. The minimum connected dominating set is the one with the minimum number of vertexes. The minimum connected dominating set problem (MCDSP) is also NP-complete.

This research was supported by Ministry of education and science of Republic of Serbia, Grant III-44006.

The MCDSP has gained popularity due to its close connection to the mobile ad hoc networks (MANETs) and sensor grids. In practical problems that can be transformed to the MCDSP there is usually no need to get the optimal solution, near-optimal solutions are sufficient in most cases.

In this paper we introduce an improved ACO algorithm for the MCDSP. The rest of the paper is organized as follows. In the next section we present different approaches to the MCDSP. In the third section a greedy algorithm for solving the MCDSP is introduced. In the fourth section we present the implementation of the ACO for the MCDSP. In the fifth section we explain our approach to avoid stagnation in ACO using a pheromone correction strategy and our method of selecting the initial vertexes. In the last section, we analyze and compare the use of pure ACO and its combination with pheromone correction on standard benchmark problems and generated examples for the MCDSP.

2. Minimum Connected Dominating Set Problem (MCDSP)

Different methods have been developed to find near-optimal solutions for the MCDSP. There are two main directions in developing algorithms for solving this problem: centralized and distributed, each of them closely connected with the type of application they are used for. In this article we focus on centralized algorithms.

Several heuristics and appropriate greedy algorithms have been developed for the MCDSP. Some of them are one-step [25] or two-step [6], [22], [12] growing techniques, or pruning-based greedy algorithms [4], [5]. A multi-step collaborative cover heuristic approach has been presented in [23]. The MSDSP has also been solved using a combination of simulated annealing and taboo search [24], neural networks [13] and parameterized approximation [10].

The ant colony optimization (ACO) is a meta heuristic that has been developed by Doringo for the traveling salesman problem [9]. ACO and other evolutionary algorithms have been proven to be effective on a wide range of combinatorial and continuous optimization problems [1], [19], [3], [2], [27]. Previously, ACO has been applied to the MDSP with great success [14], also on its weighted version [17]. For implementation of a network cluster presented as a MCDSP, a two step ACO approach was used [31]. As the first step a dominating set is created and next, as the second step, new vertexes are added to make it connected. The effectiveness of the ACO has been improved by use of different types of hybridization, like combining ACO with GA [20], [18] or differential evolution (DE) [32].

In this article we present an implementation of the ACO algorithm for the MCDSP. In our ACO implementation, we use a one-step approach applying the heuristic proposed by Guha and Khuller [12]. This approach was avoided in article [31] because of fear of early trapping in local optima and a more complex one was chosen. We propose to overcome this problem by introducing a method for avoiding early stagnation. We use a pheromone correction strategy (PCS), similar to the one used in our article [15], to direct the ant colony to

areas were good solutions are more likely. The idea of this approach is to update the pheromone trail used in ACO based on a heuristic that determines the desirability of vertexes in the solution, depending on the properties of the currently best found solution. We further improve the effectiveness of this method by implementing a good procedure for setting initial conditions of the algorithm. In our tests we show that our method is a good choice compared to existing methods and that the use of ACO combined with pheromone correction strategy has significantly better performance than the standard max-min ant system (MMAS) [26] version of ACO for this problem.

3. Greedy Algorithms for the MCDSP

There are two possible approaches to create a greedy algorithm for the MCDSP. The first one is to use a one-step approach in which the solution is constructed using only one heuristic. Another approach is represented by two-step greedy algorithms. In that case an intermediate problem like the MDSP or the maximal independent set is solved first, and at the second stage obtained solution is converted to the solution for the MCDSP as in articles [12], [6], [22]. Two-step methods usually give better results, but at the cost of being more complex for implementation. The improved results are a consequence of less constrained selection of new vertexes at the first stage. Using this type of algorithm as a base for ACO does not come natural since a separate ACO has to be developed for each stage of the algorithm.

Because of the problems mentioned, we propose a one-step greedy algorithm as a base for our ACO implementation. We have chosen to use the first greedy algorithm given by Guha and Khuller [12]. The idea of this approach is the following: we start with an initial vertex $v_0 \in V$ with the highest degree. The degree of a vertex v is the number of edges that v is incident to. Now, v_0 is the root of the tree T. At each step we pick a vertex w, which is a neighbor of some vertex v in T, that covers the highest number of uncovered vertexes. We call a vertex v covered if $v \in T$, or there exists vertex $w \in T$ for which $(v, w) \in E$. We repeat this process until all vertexes in G are covered.

To implement this greedy algorithm we need to be able to easily distinguish between neighboring, covered and uncovered vertexes. We accomplish this by using the following process. Initially all vertexes are colored white. When a new vertex is added to T it is colored black. We mark all its neighbors that are not already in T with the gray color. In the next step we select a gray colored vertex that is connected to the highest number of white vertexes. The algorithm is finished when all of the vertexes have been colored. An illustration of this algorithm is given in Fig. 1.

As noticed by Guha and Khuller [12], this type of heuristic for the greedy algorithm is easily trapped in local optimal solutions due to its short-sightedness. Because of this, more complicated algorithms have been created. Guha and Khuller have used the same approach, but instead of using single vertexes, they used pairs of them. In article [25] a heuristic that tracks the number of

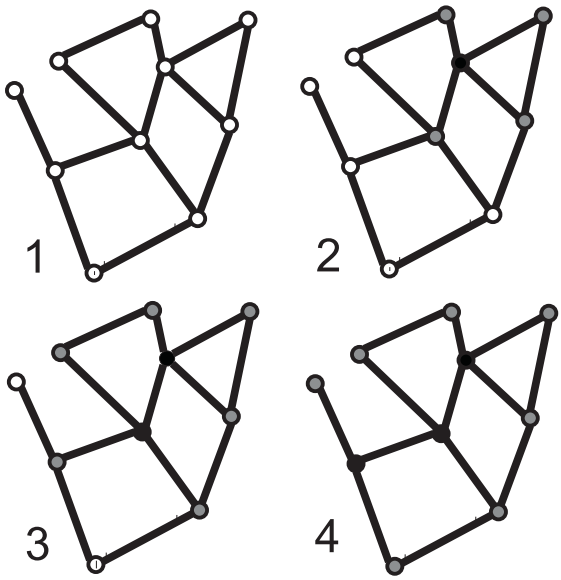


Fig. 1. Example of creating a connected dominating set using the greedy heuristic: 1) Input graph, 2) Initial step 3,4) Further steps in the algorithm

black and gray vertexes, and the number of separate black sections is used in the greedy algorithm. In [4] and [5] a greedy pruning-based approach is used where the least important vertex is removed from the dominating set. All these algorithms have a more complicated and slower implementation. We show that shortcomings of the mentioned first simple heuristic are greatly reduced when it is combined with ACO and our improvements.

4. Implementation of ACO for the MCDSP

In the ACO implementation for the MCDSP there are significant differences compared to its implementation for the traveling salesman problem (TSP). In the case of TSP the solution is a permutation of the set of all the cities; contrary to this for the MCDSP the solution is a subset of the set of graph vertexes where the order is unimportant. The heuristic function for the TSP is static because it represents the distance between cities. For the MCDSP the heuristic function is the number of white neighbors (not yet covered), which is dynamic because more vertexes are marked black or gray as new vertexes are added to the solution subset. Finally, in the case of TSP at each step all the non visited vertexes are potentially selected, while in the case of MCDSP only the vertexes marked gray are considered. These three differences affect the basic algorithm in the following way: the ants leave the pheromone on vertexes instead of edges, the heuristic function is dynamically updated and potential candidates have to be

tracked. Such variant of ACO with dynamic heuristic and a solution that consists of a subset instead of a permutation have also been used for solving the set partitioning [7], minimum vertex cover [15], set covering [21] and maximum clique [11] problems.

When implementing ACO, we first need to represent the problem in a way that makes simple the dynamic calculation of the heuristic function. This can be done in the following way. Initially, for each vertex i the value of the heuristic function η_i^0 is it's degree, or in other words, the number of connections that it has. Three sets are then created: white W^0 that initially holds all the vertexes and two empty sets, B^0 for black and Gr^0 for gray vertexes. As mentioned before, the heuristic is dynamic and it has to be updated as new vertexes are added to the result set. If at step j vertex v is added, all it's neighbors have their degree decreased by one giving the new heuristic function η^j. At this step we also move vertex v from the Gr^j to B^j, and all its neighbors from W^j to Gr^j.

To define ACO algorithm for a problem, three parts need to be defined: ant transition rule, global update rule and local update rule. We start by defining the transition rule using heuristic function η^i in the following equation:

$$p_j^k = \begin{cases} 0 & , j \notin Gr_k \\ prob_j^k & , j \in Gr_k \end{cases} \tag{1}$$

$$prob_j^k = \begin{cases} 1 & , q > q_0 \ \ \& \ \ j = \arg\max\limits_{i \in Gr_k} \tau_i \eta_i^k \\ 0 & , q > q_0 \ \ \& \ \ j \neq \arg\max\limits_{i \in Gr_k} \tau_i \eta_i^k \\ \frac{\tau_j \eta_j^k}{\sum_{i \in Gr^k} \tau_j \eta_i^k} & , q \leq q_0 \end{cases} \tag{2}$$

In Equation (2) parameter q_0 is used to define exploitation/exploration rate. Connected to it, q is a random variable upon which the next selection depends. Unlike the TSP transition rule, the selection does not depend on which vertex was added last to the current solution, but only on the current state of the graph. That is why τ_i is used instead of τ_{ij} for pheromone trail, and η_i^k instead of η_{ij} for the heuristic function. To fully specify the ACO algorithm, it remains to define the global (when ants finish their paths) and the local (when an ant chooses a new vertex) update rules.

$$\Delta\tau_i = \begin{cases} 0 & , i \notin V' \\ \frac{1}{|V'|} & , i \in V' \end{cases} \tag{3}$$

In Equation (3) $\Delta\tau_i$ is quality measure of the best global solution subset V' that contains vertex i ($|V'|$ is the number of vertexes in V'). It is used when the global update rule in Equation (4) is defined. Parameter p is used to set the influence of a newly found solution on the pheromone trail.

$$\tau_i = (1 - p)\tau_i + \Delta\tau_i \tag{4}$$

We wish to emphasis that $\Delta\tau_i$ is equal to zero for most of the vertexes, which means that the pheromone will be falling to zero for points that are not part of the global best solution.

The formula for the local update rule has the standard form

$$\tau_i = (1 - \varphi)\tau_i + \varphi\tau_0 \tag{5}$$

The quality measure of the solution acquired by the greedy algorithm (where the vertex with the best ratio of vertex degree and weight is selected) is taken for the value of τ_0. Parameter φ is used to specify the strength of the local update rule.

5. Avoiding Stagnation in ACO for the MCDSP

When ACO algorithm with the heuristic approach given by Guha and Khuller [12] is used for the MCDSP, there is a strong possibility of getting trapped in local optima. There are two main reasons for this. The first one is that this is a standard problem with ACO due to the way the pheromone matrix is created. The second one is induced by the way Guha and Khuller's greedy algorithm, which is a base for ACO implementation, works where the initially selected vertex has a very strong influence on the final result.

5.1. Pheromone Correction Strategy

We first focus on a way to avoid the problems caused by updating of the pheromone matrix. The basic approach to avoid stagnation in ACO is to use the MMAS version of ACO, in which an extra constraint is added which requires that all pheromone values are bounded, $\tau_i \in [\tau_{\min}, \tau_{\max}]$. In our case this is very important because our update rule can lower the minimum value of the pheromone very close to zero and inflicted vertexes will practically never be selected. The problem with MMAS is that for keeping the search greedy enough $\tau_{\min}$ has to be very small but the search will never be intensified after the pheromone for a vertex has reached $\tau_{\min}$.

Another interesting approach is combing ACO with the minimum pheromone threshold strategy (MPTS) as proposed in article [29]. The idea of the MPTS is to intensify search around vertexes that have been rarely selected. This is done by adding a minimum threshold value τ_{mt} that is bounded $\tau_{min} < \tau_{mt} < \tau_{max}$. In the beginning τ_{mt} is set to some initial value and then adjusted during the search, depending on the performance. Threshold τ_{mt} is used for updating the pheromone trail. When the search is conducted, values in the pheromone trail τ_i are compared to the τ_{mt} and if τ_i is lower than τ_{mt}, than τ_i is changed to some significantly higher value. In our experiments this approach proved to be efficient for small graphs, but for larger problems the search would not be greedy enough and would give results that are of lower quality than ones acquired by the MMAS version of ACO.

To improve the performance of ACO we implemented a pheromone correction strategy similar to the one used for minimum weight vertex cover problem (MWVCP) [15]. The idea of this approach is to change the pheromone

matrix by analyzing some of the properties of the best found solution. More precisely, when the search for a better solution becomes stagnant we update the pheromone matrix. We do this by using a simple heuristic function that describes the desirability of a vertex in the solution. For example, a vertex that is part of the solution and does not cover any vertexes solely by it self is not very desirable. For an undesirable vertex in the solution we greatly decrease the value of the pheromone and as a consequence, that vertex is not often chosen as a part of the solution in the following steps of the algorithm.

We have adapted this approach for the MCDSP. First, let us define $\eta(v, V')$ as the number of vertexes that vertex v, which is part of the best found solution V', solely covers.

$$Sus = \frac{1}{1 + \eta(v, V')} \tag{6}$$

In Equation (6) we have defined Sus as the undesirability of a vertex in the solution. The next step in the pheromone correction strategy is to select a random number RK of vertexes which solely cover the smallest number of vertexes. For each vertex i in the solution the probability of it being selected for pheromone correction is:

$$p_i(selected) = \frac{RK - RankSus(i, V')}{RK} \tag{7}$$

In Equation (7) instead of using the value of Sus for vertexes, we used $RankSus$ which represents their rank by undesirability . RK is the maximum number of vertexes that are considered for correction. The final step is to lower the pheromone trail for the selected vertexes:

$$\begin{array}{l} \forall i \in Selected \\ \tau_i = \delta\tau_i \end{array} \tag{8}$$

The use of $Sus(v, V')$ as a measure of desirability is not fully effective because the same group of vertexes would be repetitively selected until a better solution set was found. Because of this we introduce an improved desirability criterion:

$$CorSus(i, V') = \ Sus(i, V') \ * ExSusepect(i) \tag{9}$$

The improvement consists of tracking which vertexes have already been selected and preferring the selection of new vertexes. To do this, a new array $ExSuspect$ is introduced with elements initially set to 1. If vertex i is selected, the following correction is done:

$$\begin{array}{l} 0 < \lambda < 1 \\ ExSuspect(i) = ExSuspect(i) * \lambda \end{array} \tag{10}$$

This type of approach in which the pheromone value has been greatly decreased for some vertexes that are part of the best solution has been applied

to the MWVCP with good results [15]. The ant colony in the following steps of the algorithm avoids using these vertexes when creating new solutions. This approach however, does not give good results when extended to graph covers that also need to be connected. The problem is that when a vertex is removed, it is highly likely that it will leave the remaining vertex set disconnected. In the following steps it is hard for the ants to create a new good solution avoiding the removed vertexes due to the connectivity problem. Because of this a new type of correction is added, which is used to make it easier for new solutions to be constructed. This is done by increasing the pheromone values at vertexes that are not a part of the best found solution but are highly likely to appear in new good solution. We will consider a vertex that is not part of the solution, but covers many of the vertexes in the best solution, desirable to appear in good solutions.

Now we define a method for pheromone correction for vertexes that are not part of the best solution. First, let us define $Des(v, V')$ as the number of vertexes that are a part of the best found solution that $v \notin V'$ is connected to. The next step in the pheromone correction strategy is to select a random number RK' of vertexes which cover the greatest number of vertexes that are in the best found solution or in other words, have the greatest value of Des. For each vertex i not in the solution the probability of being selected for pheromone correction is:

$$p_i(selected) = \frac{RK' - RankDes(i, V')}{RK'} \tag{11}$$

In Equation (11) instead of using the value of Des for vertexes, we used $RankDes$ which represents their rank by desirability. RK' is the maximum number of vertexes that are considered for correction. The final step is to correct the pheromone value pheromone for the selected vertexes by increasing the value of pheromone:

$$\begin{matrix} \forall i \in Selected \\ \tau_i = \frac{\tau_{max} + \tau_{min}}{2} \end{matrix} \tag{12}$$

For vertexes for which the pheromone values will be increased we also track how often they are selected with the array $ExSuspect$ and use a new corrected desirability function $CorDes$ in the same way as for vertexes that are a part of the solution.

Finally, we need to define a stagnation criteria for recognizing if the search has been trapped in a local minimum. The criterion used is that there has been no improvement in the solution in n iterations of the ant colony. In our implementation we use separate values n_1 and n_2 for the two pheromone correction methods.

5.2. Initial Vertex Selection

When the starting point for creating an ACO algorithm for the MCDSP is Guha and Khuller's greedy algorithm, the performance is extremely influenced by the

vertex that is initially selected. This is because the solution set slowly grows from the initial vertex through its neighbors. As previously mentioned, the heuristic function is dynamic, so the previously selected vertexes not only affect the potential candidates but also which one of them will be selected. This way the initially selected vertexes have a snowball effect on the final solution. In the case of the TSP this problem is also present but it is less severe and can be solved by selecting the first vertex at random, out of all the vertexes in the graph since they all participate in the best solution. In our case this is not a good approach because only a relatively small number of vertexes are part of the best solution so the search becomes too wide. In the case of MWVCP [15] where the solution is also a small subset of V, we selected a random vertex of the best solution. However, if we choose the initial vertex for MCDSP in this way the search becomes too narrow. This is because in the case of MWVCP the previous steps only affect the heuristic function but in the case of MCDSP the candidate list is also affected.

We try to balance these two approaches in the following way:

$$InitVertex = \begin{cases} Random(V') & , s < s_0 \\ Random(V, \tau) & , s \geqslant s_0 \end{cases} \tag{13}$$

In Equation (13) s is a random variable on which the type of selection depends, s_0 is a fixed parameter that defines how often the initial vertex will be selected from the global best solution V' or from all the vertexes in V. In case it is selected from V, the probability distribution is only dependent on the pheromone trail corresponding to vertexes.

5.3. Our Improved ACO Algorithm for the MCDSP

The recapitulation of the key elements of our improved ACO algorithm for the MCDSP is:

- ACO algorithm for the MCDSP is implemented with necessary adjustments considering that for the MCDSP solution is a subset of the set of graph vertexes where the order is unimportant and that the heuristic function is dynamic. That affects the basic algorithm in a way that the ants leave the pheromone on vertexes instead of edges, the heuristic function is dynamically updated and potential candidates have to be tracked.

- The mentioned ACO algorithm is based on the first greedy algorithm given by Guha and Khuller [12]. It starts with an initial vertex $v_0 \in V$ with the highest degree as the root of the tree T. At each step a vertex w is picked, which is a neighbor of some vertex v in T, that covers the highest number of uncovered vertexes. This process is repeated until all vertexes in G are covered.

– ACO algorithm for the MCDSP based on Guha and Khuller's greedy algorithm is strongly influenced by the vertex that is initially selected because the solution set slowly grows from the initial vertex through its neighbors. We introduce modification that narrows the selection to vertexes that belong to the global best solution, but not always, according to Equation (13).

– When stagnation is detected, search has to move to, at that moment, less promising areas. Rather than using more standard method of increasing the pheromone level for vertexes that currently do not belong to the best found solution, we decrease the pheromone level for, by defined criteria, undesirable vertexes in the best found solution. This novel approach improves leaving local optima in the directions that lead to better solutions.

– The previous step, very successful for some other problems [15], creates some problems when unmodified applied to graph covers that also need to be connected. The problem is that when a vertex is removed, it is highly likely that it will leave the remaining vertex set disconnected. Because of this a new type of correction is added, which is used to make it easier for new solutions to be constructed. This is done by increasing the pheromone values at vertexes that are not a part of the best found solution but are, by defined criteria, highly likely to appear in new good solution.

The program for our experiments was written in C#, using the framework from article [16]. The program implements the following pseudo code

```
Reset Graph Info
Reset Solution for all Ants
Select Initial Vertex for all ants

 while  (! AllAntsFinished)
   for All Ants
     if(AntNotFinished)
       begin
         add new vertex A to solution based on probability
         correct ant's cover graph data
         calculate new set of candidates
         calculate new heuristic
         local update rule for A
       end
     end for
   end while

 Compute DeltaTauI
 Compute TauI
```

```
If(Iteration_NoChange % n1)
  Use CorSus for Pheromone Correction
If(Iteration_NoChange % n2)
  Use CorDes for Pheromone Correction
```

6. Test and Results

We have conducted two types of tests. In the first type we analyze the effectiveness of our method on benchmark data sets with existing solutions. In the second group of tests we generate problem instances as proposed in article [14] that correspond to ad hoc network clustering problems.

The ACO algorithm is implemented in its MMAS version. For both, ACO with and without pheromone correction, we conducted ten separate colony simulations and compared average solutions and standard deviations. All the colonies had the following parameters: $q_0 = 0.9$ specifies the exploitation/exploration rate, $p = 0.1$ and $\varphi = 0.1$ specify the global and local update rules. These are the standard values used by most authors and after some testing we decided that there is no need to change them. The value of the parameter that defines initial vertex selection is $s_0 = 0.2$. This parameter is specific for our method and was determined empirically after significant number of tests. The parameters used for our pheromone correction had the following values: coefficient for the pheromone correction $\delta = 0.0001$, the maximum number of selected vertexes $RK = \frac{|V|}{s}$ where s is a random number from the interval [2,10] and $\lambda = 0.9$. We determined these parameters in [15] and after some testing determined that the same values are appropriate for this problem. The stagnation parameters had the following values: $n_1 = 20$ and $n_2 = 40$. These values were empirically proven to balance two corrections specific for our method. In our tests we used 10 colonies for both, ACO and ACO with pheromone correction strategy. In both cases we used random seeds with values from 0 to 9.

We have tested our method on benchmark data sets that have been used on the Tenth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'09) [28]. The maximum number of iterations for a colony was 350 which means that 3500 solutions have been created. We compare the quality of solutions achieved by our ACO combined with a pheromone correction strategy to standard MMAS ACO using the basic version of Guha and Khuller's heuristic, to pure greedy algorithm and to known best benchmark results from the LPNMR'09. These results are in Table 1.

In Table 1 we only give the results for problem instances that have had a satisfactory solution (the solution is known) given in the LPNMR benchmark. The best found solution for the ant colonies, which is commonly shown, does not appear in the table due to the fact that both ACO algorithms have achieved the best given solution in all the benchmark examples. We first notice that the basic greedy algorithm of Guha and Khuller performs poorly and gives the average error of 126% compared to the best solution. The MMAS variation of ACO gives results that on average have 8.5% error. This shows that the use of

Table 1. Comparison of LPNMR, Greedy 1, simple ACO and ACO combined with MPTS

Problem	LPNMR	Greedy	ACO MMAS			ACO with PCS		
Dimensions	Result		Average	St.Dev.	t	Average	St.Dev	t
40*200	5	10	5.8	0.60	3.2	5.3	0.45	4.1
45*250	5	15	5.8	0.40	3.5	5.5	0.50	4.3
50*250(1)	8	15	8.1	0.54	4.8	8.0	0.00	6.1
50*250(2)	7	17	7.5	0.50	5.0	7.1	0.30	6.5
55*250	8	20	8.8	0.98	5.6	8.3	0.45	7.3
60*400	7	15	7.0	0.00	6.1	7.0	0.00	9.1
70*250	13	32	14.2	0.74	11.0	13.9	1.04	13.5
80*500	9	20	10.0	0.44	12.1	9.8	0.40	16.9
90*600	10	19	10.9	0.83	14.0	10.6	1.01	17.3
Average	8.00	18.11	8.68			8.34		

ACO, with careful selection of the initial vertex, manages to overcome the shortsightedness of the underlying greedy method. Finally, the results that have been archived by adding the pheromone correction strategy to ACO manages to improve the results even further to have an average error of 4.2%. Standard deviation is also improved in most cases. Columns marked with t report computational times in seconds for ten runs. They should be used only for coarse comparison since they include hard disk time, no optimization of the algorithm was attempted and it was written in C#.

As an illustration of the effectiveness of this method we give a comparison with results for the problem viewed as decision problem achieved by Answer Set Programming (ASP), Propositional Satisfiability (SAT) and Constraint Programming (CP) that are given on the LPNMR'09 web site. The benchmark test set consists of 21 problems of different sizes, and for each it is requested to answer if a solution of a certain number of vertexes exists. For each of the test examples we have conducted two colony runs with a fixed number of iterations (350), and we check if any of the colonies has found a solution with the requested number of vertexes; if it has the problem is satisfied, otherwise it is not. The test have been done on similar hardware (ours slightly better): at LPNMR'09 Dell OptiPlex 745, 1 CPU with 2 cores: GenuineIntel Intel(R) Core(TM)2 CPU 6600 @ 2.40GHz 4 GB RAM, and in our case Dell OptiPlex 755, 1 CPU with 2 cores: GenuineIntel Intel(R) Core(TM)2 CPU E8500 @ 3.16GHz 3 GB RAM. The software used at LPNMR'09 was created in C++ and our application was made using C# which gives them a speed advantage. Our method had successfully solved all the problem instances and for that it needed 52 seconds. In comparison to this, the best method from LPNMR'09 has solved all the problems in 36 seconds, and the following ones needed 128, 169, 316, 465 and 535 seconds. Although the comparison is not fully accurate it still shows that our method is very competitive.

In our second group of tests we generated graphs in the same way as proposed by Chen [14]. The graphs are generated in the following way. In some fixed area $N * N$ a random number of points are selected with a uniform distribution which represent the nodes of our graph. If the distance between two nodes i and j is smaller that some value R than edge (i, j) is a part of our graph. We have generated problems with different number of nodes and different edge densities and used them to compare ACO and ACO with a pheromone correction strategy. We use the same parameters for ACO as before, except for the maximum number of iterations for a colony which is now 5000 due to the increased size of the problems. We can see the results in Table 2.

For each of the 41 test instances we compared the best found solution and the average solution for ACO and ACO combined with pheromone correction. We first wish to point out that the basic greedy algorithm performs poorly for larger problem instances. Both ACO approaches improve the minimal solution 2-3 times compared to the greedy algorithm. ACO combined with a pheromone correction strategy improved the best found solution in 18 cases and decreased its quality in only 3 cases. When the average solution is observed the addition of a pheromone correction strategy improved the result quality in 33 cases and decreased its quality in 6 cases. The advantages of using the PCS are greater in the case of small and medium problem instances. We explain this by the fact that the PCS parameter values have been chosen from analyzing the behavior of the algorithm for small problem instances. We believe that the same level of improvement can be archived with a better choice of parameters.

7. Conclusion

In this paper we have presented an ant colony optimization algorithm for the minimum connected dominating set problem. Our implementation is fast and simple one-step ACO method based on a greedy heuristic where our pheromone correction strategy and special attention to the initial condition of the ACO overcome shortcomings of that heuristic. The tests on standard benchmark data as well as on standard generated examples have shown that our algorithm generates good solutions compared to other state of the art algorithms. Moreover, the execution time is favorable compared to the results obtained on 10^th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'09) benchmark data sets. This is important since such solutions are usually used in MANETs and the speed of execution is more important than optimality. We used successfully the similar strategy to improve ACO for the MWVCP and another version for the TSP so we can consider that our pheromone correction strategy is a rather general method of improving ACO. Future research may include additional tuning for larger examples and use of different pruning-based greedy heuristics as in [4], [5]. They are more complex for implementation but natural for the ACO since these are one-step algorithms that much less depend on the initial vertex selection. Some recent improvements in greedy algorithms [30], [8] can also be included in the future research.

Table 2. Comparison of ACO and ACO combined with pheromone correction on different MCDSP instances

Area(N*N) Nodes	R	Greedy	ACO		ACO + PCS	
			Min	Avg	Min	Avg
400	60	48	20.0	21.6	19	**21.2**
80	70	33	16	17.0	**15**	**16.2**
	80	35	12	14.0	12	**13.1**
	90	41	11	11.8	11	**11.6**
	100	23	8	9.0	8	**8.9**
	110	25	8	8.5	8	8.5
	120	17	7	7.5	7	**7.2**
600	80	38	23	24.7	**22**	**23.6**
100	90	40	22	23.8	**21**	**23.6**
	100	38	17	20.0	17	**19.0**
	110	35	15	17.2	15	**16.8**
	120	36	15	16.2	**14**	**15.5**
700	70	96	46	50.7	46	**49.6**
200	80	89	41	**43.7**	41	43.9
	90	84	34	36.0	**33**	**35.7**
	100	75	28	**30.8**	28	31.0
	110	70	23	27.4	**22**	**26.4**
	120	68	21	23.6	21	**23.4**
1000	100	96	46	50.7	46	**49.6**
200	110	92	43	44.9	**42**	**44.8**
	120	82	37	39.9	37	**39.8**
	130	91	32	34.7	32	**34.9**
	140	76	30	31.3	**29**	31.3
	150	83	28	29.6	**26**	**28.8**
	160	86	**24**	26.6	25	**26.5**
1500	130	158	60	64.5	60	**64.3**
250	140	144	53	57.2	**52**	**57**
	150	170	51	54.9	51	**54.4**
	160	151	47	50.5	**45**	**49.8**
2000	200	178	55	**58.6**	**52**	58.8
300	210	151	51	53.5	**50**	**52.8**
	220	140	47	48.9	**45**	**48.4**
	230	166	44	47.5	44	**46.9**
2500	200	198	79	82.0	79	**81.5**
350	210	185	75	79.1	**74**	**78.2**
	220	205	**68**	**72.6**	69	73.8
	230	193	66	69.2	66	**68.9**
3000	210	259	99	**101.6**	**98**	104.0
400	220	225	**88**	**95.4**	91	97.6
	230	205	86	91.4	86	**90.3**
	240	210	82	85.8	**80**	**84.1**

Acknowledgment. Authors thank anonymous reviewers for useful comments that helped improve the quality of this paper.

References

1. Abbaspour, R.A., Samadzadegan, F.: An evolutionary solution for multimodal shortest path problem in metropolises. Computer Science and Information Systems 7(4), 789–811 (2010)
2. Bacanin, N., Tuba, M.: Artificial bee colony (ABC) algorithm for constrained optimization improved with genetic operators. Studies in Informatics and Control 21(2), 137–146 (2012)
3. Brajevic, I., Tuba, M.: An upgraded artificial bee colony algorithm (ABC) for constrained optimization problems. Journal of Intelligent Manufacturing (published Online First), DOI: 10.1007/s10845–011–0621–6 (2012)
4. Butenko, S., Cheng, X., Oliveira, C., Pardalos, P.: A new heuristic for the minimum connected dominating set problem on ad hoc wireless networks. In: Recent Developments in Cooperative Control and Optimization. pp. 61–73. Kluwer Academic Publishers (2004)
5. Butenko, S., Oliveira, C., Pardalos, P.: A new algorithm for the minimum connected dominating set problem on ad hoc wireless networks. In: CCCT'03. pp. 39–44. International Institute of Informatics and Systematics (IIIS) (2003)
6. Cheng, X., Ding, M., Chen, D.: An approximation algorithm for connected dominating set in ad hoc networks. In: Proc. of International Workshop on Theoretical Aspects of Wireless Ad Hoc, Sensor and Peer-to-Peer Networks (TAWN) (2004)
7. Crawford, B., Castro, C.: Ant colonies using arc consistency techniques for the set partitioning problem. In: Professional Practice in Artificial Intelligence. pp. 295–301. Springer, Boston (2006)
8. Das, A., Mandal, C., Reade, C., Aasawat, M.: An improved greedy construction of minimum connected dominating sets in wireless networks. In: 2011 IEEE Wireless Communications and Networking Conference (WCNC). pp. 790–795. IEEE (2011)
9. Dorigo, M., Gambardella, L.M.: Ant colonies for the travelling salesman problem. Biosystems 43(2), 73–81 (July 1997)
10. Downey, R.G., Fellows, M.R., McCartin, C., Rosamond, F.A.: Parameterized approximation of dominating set problems. Information Processing Letters 109(1), 68–70 (2008)
11. Fenet, S., Solnon, C.: Searching for maximum cliques with ant colony optimization. In: Applications of Evolutionary Computing. pp. 291–302. Springer-Verlag, Berlin/Heidelberg (2003)
12. Guha, S., Khuller, S.: Approximation algorithms for connected dominating sets. Algorithmica 20(4), 374–387 (1998)
13. He, H., Zhu, Z., Makinen, E.: A neural network model to minimize the connected dominating set for self-configuration of wireless sensor networks. IEEE Transactions on Neural Networks 20(6), 973–982 (June 2009)
14. Ho, C.K., Singh, Y.P., Ewe, H.T.: An enhanced ant colony optimization metaheuristic for the minimum dominating set problem. Applied Artificial Intelligence 20(10), 881–903 (2006)
15. Jovanovic, R., Tuba, M.: An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem. Applied Soft Computing 11(8), 5360–5366 (December 2011)

16. Jovanovic, R., Tuba, M., Simian, D.: An object-oriented framework with corresponding graphical user interface for developing ant colony optimization based algorithms. WSEAS Transactions on Computers 7(12), 1948–1957 (2008)
17. Jovanovic, R., Tuba, M., Simian, D.: Ant colony optimization applied to minimum weight dominating set problem. In: Proceedings of the 12th International conference on Automatic control, modelling and simulation. pp. 322–326. ACMOS'10, World Scientific and Engineering Academy and Society, Stevens Point, Wisconsin, USA (2010)
18. Jun-Qing Li, Q.K.P., Xie, S.X.: A hybrid variable neighborhood search algorithm for solving multi-objective flexible job shop problems. Computer Science and Information Systems 7(4), 907–930 (2010)
19. Kratica, J., Kostic, T., Tosic, D., Dugosija, D., Filipovic, V.: A genetic algorithm for the routing and carrier selection problem. Computer Science and Information Systems 9(1), 49–62 (2012)
20. Lee, Z.J., Su, S.F., Chuang, C.C., Liu, K.H.: Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. Applied Soft Computing 8(1), 55–78 (2008)
21. Lessing, L., Dumitrescu, I., Sttzle, T.: A comparison between ACO algorithms for the set covering problem. In: LNCS 3172, Springer. pp. 1–12 (2004)
22. Min, M., Du, H., Jia, X., Huang, C.X., Huang, S.C.H., Wu, W.: Improving construction for connected dominating set with steiner tree in wireless sensor networks. Journal of Global Optimization 35(1), 111–119 (2006)
23. Misra, R., Mandal, C.: Minimum connected dominating set using a collaborative cover heuristic for ad hoc sensor networks. IEEE Transactions on Parallel and Distributed Systems 21(3), 292–302 (June 2010)
24. Morgan, M., Grout, V.: Metaheuristics for wireless network optimisation. In: The Third Advanced International Conference on Telecommunications, AICT 2007. p. 15 (May 2007)
25. Ruan, L., Du, H., Jia, X., Wu, W., Li, Y., Ko, K.I.: A greedy approximation for minimum connected dominating sets. Theor. Comput. Sci. 329(1-3), 325–330 (2004)
26. Stützle, T., Hoos, H.H.: Max-min ant system. Future Gener. Comput. Syst. 16(9), 889–914 (June 2000)
27. Tuba, M., Brajevic, I., Jovanovic, R.: Hybrid Seeker Optimization Algorithm for Global Optimization. Applied Mathematics and Information Sciences, 7(3), 867–875 (2013)
28. URL: The second answer set programming (asp) competition: Submitted benchmarks (2009), http://dtai.cs.kuleuven.be/events/ASP-competition/encodings.shtml
29. Wong, K.Y., See, P.C.: A new minimum pheromone threshold strategy (MPTS) for max-min ant system. Applied Soft Computing 9(3), 882–888 (June 2009)
30. Yang, D., Wang, X.: Greedy Algorithms for Minimum Connected Dominating Set Problems. In: Proceeding of the 10th International Conference on Intelligent Technologies. pp. 643–646 , Guangxi Normal Univ., Guilin, Peoples Republic of China, (December 2009)
31. Zhang, C., Xu, Q.: Clustering approach for wireless sensor networks using spatial data correlation and ant-colony optimization. In: Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing - Volume 01. pp. 538–541. IEEE Computer Society, Washington, DC, USA (2009)
32. Zhang, X., Duan, H., Jin, J.: Deaco: Hybrid ant colony optimization with differential evolution. In: IEEE Congress on Evolutionary Computation. pp. 921–927. IEEE Computer Society (2008)

Raka Jovanovic is a Ph. D. candidate at the University of Belgrade, Faculty of Mathematics where he also received B.S. and M.S. degrees in Computer Science. He worked as a research assistant/associate at the Institute of Physics, University of Belgrade and was employed as a research associate at Texas AM University at Qatar. His research interests include Optimization problems, Data compression, Image processing, Numeric simulation and Fractal imaging.

Milan Tuba is Professor of Computer Science and Provost for mathematical, natural and technical sciences at Megatrend University of Belgrade. Before that he was associate professor at Faculty of Mathematics, University of Belgrade and assistant professor of Electrical Engineering at Cooper Union, New York. He received B. S. in Mathematics, M. S. in Mathematics, M. S. in Computer Science, M. Ph. in Computer Science, Ph. D. in Computer Science from University of Belgrade and New York University. His research interest includes mathematical, queuing theory and heuristic optimizations applied to computer networks, image processing and combinatorial problems. Professor Tuba is the author of more than 100 scientific papers. He is coeditor or member of the editorial board or scientific committee of number of scientific journals and conferences. Member of the ACM since 1983, IEEE 1984, New York Academy of Sciences 1987, AMS 1995, SIAM 2009, IFNA 2012.

Received: September 22, 2011; Accepted: October 10, 2012.

DOI:10.2298/CSIS110804035G

Ontological Model of Legal Norms for Creating and Using Legislation

Stevan Gostojić, Branko Milosavljević, and Zora Konjović

Faculty of Technical Sciences, University of Novi Sad
Trg D. Obradovića 6, 21000 Novi Sad, Serbia
{gostojic, mbranko, ftn_zora}@uns.ac.rs

Abstract. This paper presents a formal model of legal norms modeled in OWL. It is intended for semiautomatic drafting and semantic retrieval and browsing of legislation. Most existing solutions model legal norms using formal logic, rules or ontologies. Nevertheless, they were not intended as a basis for drafting, retrieval and browsing of legislation. The proposed model formally defines legal norms using their elements and elements of legal relations they regulate. The duality between the content and the form of legislation is exploited by connecting it to the XML model of legislation based on the CEN MetaLex specification. Those models are verified by applying them to the norms contained in an existing piece of legislation and by developing a prototype application for semantic browsing of legislation that is based on the models.

Keywords: legal norms, legislation, ontology, OWL, XML, CEN MetaLex, browsing

1. Introduction

The quality of legislation and legislative drafting procedures is questionable. Drafting of legislation starting from its semantics (cf. [1]), with the semi automation of the application of legislative drafting guidelines, can improve the quality of legislation (its consistency, intelligibility and usability) and drafting procedure (its efficiency and effectiveness).

In order to make decisions, lawyers use legislation corpus as a knowledge base of legal norms and their relations, since legal norms are applied as they are formulated in legislation. Traditional legislation retrieval and browsing systems are based on text retrieval and browsing. Those systems do not solve the problem of legal rule fragmentation (the property of the legal system that legal norms which regulate one social relation or elements of one legal norm are contained in different legislation or different elements of a piece of legislation). This property is one of the main reasons for ineffective and inefficient usage of legislation, especially by citizens who are not lawyers. The semantic retrieval and browsing of legislation, based on the meaning of the legal norms it contains, is a promising solution to this problem.

This paper proposes a formal model of legal norms used as a basis for the development of expert systems for semiautomatic drafting and semantic retrieval and browsing of legislation. It is connected with the formal model of legislation based on the CEN MetaLex specification as described in [2]. The model was specified in Web Ontology Language (OWL).

The rest of this article is structured as follows. Section 2 reviews related work. Section 3 defines basic legal concepts and describes the proposed formal model of legal norms that is based on those concepts. Section 4 gives an example of the usage of the proposed model as applied to norms contained in a specific legislation [3] and describes a prototype application used for semantic browsing of legislation. Finally, the last section gives concluding remarks and proposes directions of future research.

2. Related Work

Most commonly used formalisms for the representation of legal norms are formal logic, rules and ontologies. Some logical formalisms for their representation are described in [4], [5] and [6].

Biagioli and Grossi in [4] present a logic-based approach to legislative meta-drafting. They introduce classes of meta-data, corresponding to the specific classes of legal provisions. The provisions in the model are divided into two main families: rules (constitutive and regulative provisions) and rules on rules (modificatory provisions). The constitutive provisions lay out the components of the relevant pieces of legislation by introducing new types of entities, defining new terms or procedures, creating new institutional bodies, and attributing powers. The regulative provisions concern deontic concepts. The modificatory provisions manage the dynamics of laws. They are divided into modifications and derogations.

This formal model expressed in DL had large influence on the design of our ontology. Nevertheless, we have come to different results by introducing legal relation in our model and paying special attention to the structure of the legal system, the legal norm and the legal relation.

Sartor in [5] gives a formal reconstruction of some fundamental patterns of legal reasoning. Legal norms are represented as unidirectional inference rules that can be combined into arguments. The value of each argument (its qualification as justified, defensible, or defeated) is determined by the importance of the rules it contains. Applicability arguments, intended to contest or support the applicability of legal norms, preference arguments, purporting to establish preference relations among norms, and interpretative arguments are also formalized.

Gordon in [6] presents Legal Knowledge Interchange Format (LKIF). LKIF is an XML schema for representing theories and proofs constructed from theories. A theory in LKIF consists of a set of axioms and inference rules.

Some ontologies that model legal norms are Conceptual frame-based ontology of Law [7], FOLaw [8], LRI-Core [8], DOLCE+CLO [9], OWL Ontology of Fundamental Legal Concepts [10] and LKIF-Core [11].

Conceptual frame-based ontology of Law is constituted by three frame structures. These structures are the norm frame, the act frame and the concept-description frame. A legal-theoretical analysis has determined the form of the structures. Every norm must comprise a norm subject, a legal modality and an act description. Identified types of norms are norms of conduct, norms of competence, duty imposing, permissive, general, individual, categorical and hypothetical norms. Depending on the type of a norm (categorical or hypothetical), these elements can be supplemented with conditions of application. The aspects of the act are an agent, an act type, a modality (modality of means and manner), a setting (temporal, spatial and circumstantial aspect), a rationale (a cause, an aim and an intentionality) and a final state. The aspects of the concept description are the concept to be defined, conditions under which a concept is applicable, instances of a concept, a concept type and application provisions. Some additional elements of all three frames are the identifier (used as a point of reference for a frame), the promulgation (links a frame to its source) and the scope (limits the application range of a frame).

The focus of the Conceptual frame-based ontology of Law was on conceptual primitives used to model the legal domain, not on the formal version of the ontology nor on the development of expert systems. Therefore, the result of this research could not be directly applied to the drafting, retrieving and browsing of legislation.

FOLaw and LRI-Core ontologies were developed at the Leibniz Center for Law. FOLaw specifies functional dependencies between types of knowledge involved in legal reasoning. It distinguishes six types of knowledge. Normative knowledge is the most typical category of legal knowledge, norms express (un)desirable behavior using deontic operators permission, obligation and prohibition. Meta-legal knowledge is knowledge needed to resolve conflicts between individually applicable norms. World knowledge contains description of the behavior in the world of discourse. Responsibility knowledge establishes a relation between the violation of a norm and an agent who is responsible for its violation. Reactive knowledge specifies which reaction should be taken when the norm is violated. Creative knowledge allows the creation of social institutions and legal persons. The authors have developed a new representation and inference formalisms for the normative knowledge that are an alternative to deontic logic [12].

FOLaw is a functional ontology. It presents a legal-sociological view rather than a perspective from the law itself since it is based on the roles that the legal system plays in a society. Structural ontology of law is better suited for drafting, retrieval and browsing of legislation.

LRI-Core is written in OWL. One may distinguish many concepts in law, but not many are typical for law. These concepts are usually specializations of common sense concepts. Therefore, LRI-Core contains two levels. The more abstract level is a foundational ontology that covers concepts from physical, mental, and abstract worlds and roles. The more concrete level is a legal core

ontology. The legal core ontology is used for development of domain ontologies.

Its major objective is to provide support for developing legal domain ontologies by clarifying common conceptual denominators in the legal domain (e.g. role, norm, responsibility, etc). As such, it is too abstract with respect to the goals set out during the development of the ontology described in this paper.

Core Legal Ontology (CLO) is a result of collaboration between ISTC-CNR and ITTIG-CNR. It organizes legal concepts and relations about the physical, cognitive, social, or properly legal worlds based on formal properties defined in DOLCE. In CLO, a legal norm is a subclass of the social norm, which is expressed by a normative text, and is realized by a document. It distinguishes constitutive and regulative norms. Constitutive norms introduce new entities in the ground ontology, while regulative norms provide constraints on existing ground entities. Definitions and power-conferring rules are subclasses of constitutive norms. Regulative norms define behavior courses, and have at least one modal description as a proper part.

Although it is useful for the definition of legal domain ontologies, it is our opinion that CLO is rather heavyweight for the problem we are planning to solve and does not describe the structure of the legal system with the needed level of detail.

The OWL Ontology of Fundamental Legal Concepts has been developed under the ESTRELLA [13] project with the aim of clarifying the basic theoretical constituents of legal concepts and of contributing to enable semantic access to digital legal information. The formal language chosen to express the first version of this ontology is OWL. The first classification of legal concepts includes two main classes: norms and normative judgments. Norms state normative judgments. Norms can be unconditional, that is their judgment may not depend upon any antecedent condition. Conditional norms are distinguished into rules that make a normative judgment dependent upon sufficient conditions. Initiation rules state that a certain normative proposition starts to hold when the rule's conditions are satisfied. Termination rules state that a normative proposition ceases to hold when the rule's conditions are satisfied. Supervenience rules state that a normative proposition holds as long as the conditions are satisfied. Factor-links make a normative judgment dependent upon contributory conditions (the condition favors the judgment, but it does not determine it). A normative judgment is the proposition expressing or stating a normative fact.

The LKIF-Core ontology consists of several modules, each representing a relatively independent cluster of concepts: expression, norm, process, action, role, place, time and mereology. The concepts in these modules were formalized using OWL. It is divided into three layers: the top level, the intentional level and the legal level. The top-level clusters of the ontology provide definitions of the context in which any legally relevant fact, event or situation occurs. Modules at the intentional level include concepts and relations necessary for description of mental state and behavior of agents. At the legal level, the LKIF-Core ontology introduces a comprehensive set of legal agents and

actions, rights and powers, typical legal roles, and concept definitions that allow the expression of normative statements.

Although the most comprehensive legal ontology so far, in our opinion LKIF-Core is not suitable for the solution of the posed problem for similar reasons as CLO.

Yet another formalism for the representation of legal norms is described in [14]. Olbrich and Simon in [14] discuss visualization and formal modeling of a legally regulated process. They explicitly derive a process structure that is implicitly specified within the paragraphs themselves. The Semantic Process Language (SPL) is used to translate paragraphs into process models, since it enables articulation of language structures into executable workflow models.

Not surprisingly, the main element of all reviewed ontologies is the (legal) norm. Some of those ontologies also identified key concepts such as (legal) subjects, (legal) actions, legislation, etc.

Nevertheless, none of the reviewed ontologies identifies legal relation as a key concept in the legal domain and they do not pay special attention to the structure of the legal system, the legal norm and legal relation.

3. Ontological Model of Legal Norms

The model of legal norms presented in this paper adopts the structural view of the legal system and defines other concepts starting from the elements of the legal relation and the legal norm. It is based on the related work on modeling legal norms reviewed in Section 2 and the interpretation of legal-theoretic views presented in [15], [16] and [17].

Bearing in mind computational properties, we decided to develop a lightweight ontology suited for a particular task instead of adopting existing general-purpose ontologies. Some of the specified concepts (e.g. subject, object, act, social norm, etc.) have very general meaning and can be imported from existing foundational ontologies. One candidate is DOLCE ontology [18] because it focuses on social entities (e.g. organizations, collectives, norms, etc.) and is minimal in comparison to other foundational ontologies. Other concepts (i.e. legal norm, legal act, right, duty, etc.) have more precise (legal) meaning and can be imported from existing legal ontologies. Some candidates are CLO and LRI-Core (especially CLO because it shares many concepts such as legal norm, legal fact, legal act, legal subject with our ontology). Nevertheless, this was not our primary concern in this paper.

As noted earlier, the purpose of this model is to provide for semiautomatic drafting and semantic retrieval and browsing of legislation by exploiting duality between the form of legislation (textual formulation of a system of legal norms) represented in XML using CEN MetaLex compliant model and the content of legislation (a system of legal norms contained in it) represented in RDF using the proposed model.

That means that the scope of the model are general and abstract legal norms, abstract social relations, abstract subjects, abstract objects and legis-

lation since the legal system is a system of general and abstract legal norms, while legislation formulates a part of the legal system. Abstract terms refer to ideas or concepts. Concrete terms refer to objects or events that can be sensed. General terms refer to groups. Specific terms refer to individuals.

We used top-down approach to ontology development to identify and formally specify concepts that are essential for the description of a legal system (a system of legal norms) paying attention to criteria such as clarity, coherency and extensibility [19].

Legal concepts were modeled as OWL classes while relations between those concepts were modeled as OWL properties. OWL was used as a modeling language because of its inference semantics, open world assumption and distributed nature. Inference semantics allows the use of existing tools (OWL reasoners and RDF data stores) as the basis for the development of expert systems. We have chosen to use OWL DL sublanguage because it offers maximum expressiveness without losing computational completeness and decidability. Open world assumption is a natural state of affairs in the legal domain. This model has to be distributed since different people (or organizations) will presumably model different parts of a legal system. Furthermore, the usage of open standards promotes technical interoperability with other information systems and the usage of existing tools.

The most important classes and properties of the model are described in this section (special attention is paid to the legal relation and the legal norm as central classes of the model). They are expressed either textually using N3 notation or graphically using figures created by the Protégé tool. When using N3 notation, namespace prefixes and nonessential properties are omitted due to space constraints. The full version of the ontology can be downloaded from [20].

A subject (*Subject*) is an observer (of an object). According to [15] and [16], it can be abstract (e.g. a natural person) or concrete (Alice). Since concrete subjects are out of the scope of our model, (the concept of) a subject was modeled as an OWL class, while a natural person (an abstract subject) was modeled as an OWL instance. On another level of abstraction, (the concept of) a natural person could be modeled as an OWL class, while Alice could be modeled as an OWL instance. In that case, the natural person could be a class and an instance at the same time, although that would compromise computational completeness and decidability of the model.

One abstract subject can be a specialization or a generalization of another abstract subject (e.g. a natural person is a specialization of a person). Those relations were modeled with *specializes* and *generalizes* properties. It should be noted that built-in *rdfs:subClassOf* property could not be used because it applies to classes only.

A legal subject (*LegalSubject*) is a subject that is a part of a legal relation. In other words, it is the holder of legal capacity.

An object (*Object*) is a thing being observed (by a subject). Objects can also be abstract (a telephone number) or concrete (the +381214852426 telephone number). One abstract object can be a specialization or a generaliza-

tion of another abstract object (e.g. a telephone number is a specialization of the personal data).

A legal object (*LegalObject*) is an object that connects legal subjects into legal relations or, in other words, an asset that is allocated between legal subjects. That asset can be a physical object (e.g. land, human body, data carrier, etc.) or a mental object (e.g. intellectual property, honor, data, etc.).

A social relation (*SocialRelation*) is a relation between two or more subjects. Abstract social relations are relations between abstract subjects (e.g. love between a man and a woman). Concrete social relations are relations between concrete subjects (e.g. love between Romeo and Juliet).

Since social relations are usually organized into hierarchies, one relation can be a specialization or a generalization of another relation (e.g. being a child is a specialization of being a descendant).

The legal relation (*LegalRelation*) is a social relation (*SocialRelation*) that is regulated by a legal norm (*LegalNorm*). This is a central class of the model. The legal relation is a starting point when modeling legal norms (that will be transformed into legislation). It is also used for retrieval of norms and legislation using criteria such as regulated social relations, addressed subjects and deontic modalities. To accommodate for that use case, legal relations have elements.

The elements of the legal relation (*RelationElement*) are a right (*Right*) and a duty (*Duty*). A right is the possibility of acting according to a particular disposition that is protected by the state. A duty is the necessity of acting according to a particular disposition that is sanctioned by the state. An obligation (*Obligation*) is a duty that orders particular action. A prohibition (*Prohibition*) is a duty that forbids particular action. A competence (*Competence*) is a right to act in the interest of another legal subject, so it is a right and a duty at the same time. The elements of legal relation are shown in Figure 1.

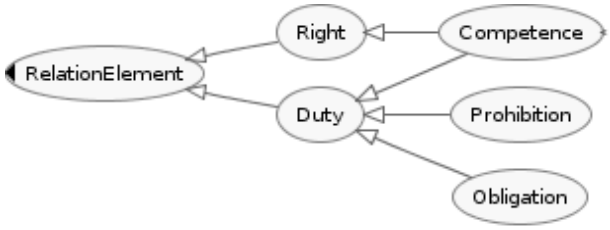


Fig. 1. The elements of a legal relation.

Legal relation elements connect legal subjects and legal objects into legal relations. A legal subject is connected with a legal relation element (its right or duty) with *has* property. Legal relation elements are connected with a legal object with *allocates* property. That way, a legal object connects the right of one subject and the duty of another subject into legal relation. Relations between legal relation, legal relation elements, subjects and object are shown in Figure 2.

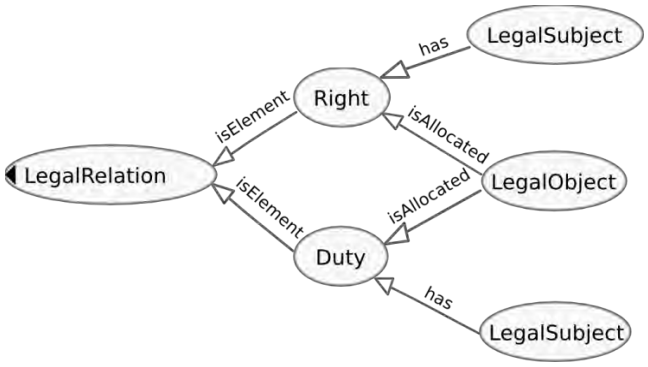


Fig. 2. The relation between legal relation and its elements.

A policy (*Policy*) determines the purpose of a legal norm, reasons why some social relations are acceptable to the society (or the state) while others are not. Usually, the purpose of the legal norm is to promote or preserve social values. Those values can also be promoted or preserved with other types of social norms. There are different types of policies: abstract policy (*AbstractPolicy)* or concrete policy (*ConcretePolicy)*, basic policy (*BasicPolicy)* or special policy (*SpecialPolicy)*, temporary policy (*TemporaryPolicy)* or permanent policy (*PermanentPolicy)*, etc. Different classes of policies are implemented with different classes of legal norms. For example, temporary policies are usually implemented with norms that have a date of repeal. Policies are shown in Figure 3.

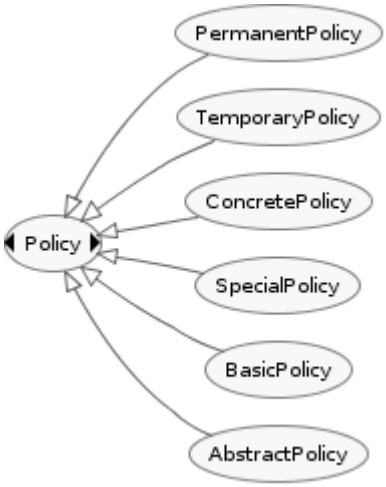


Fig. 3. The types of policies.

The policy is used for the interpretation of the meaning of legal norms that implement it. According to legislative drafting guidelines, each law is sup-

posed to explicitly state policies it implements, so judges and public officers could use textual formulations of policies to interpret and apply legal norms.

A social norm (*SocialNorm*) is a rule of conduct (or behavior) in a society. There are different kinds of social norms such as customs, moral and legal norms.

Social norms can be abstract or concrete and general or individual. Abstract norms are usually general and vice versa, but that is not always the case. A norm that pardons all prisoners for a concrete reason is a concrete and a general norm. A norm that elects a specific judge is an abstract and a specific norm.

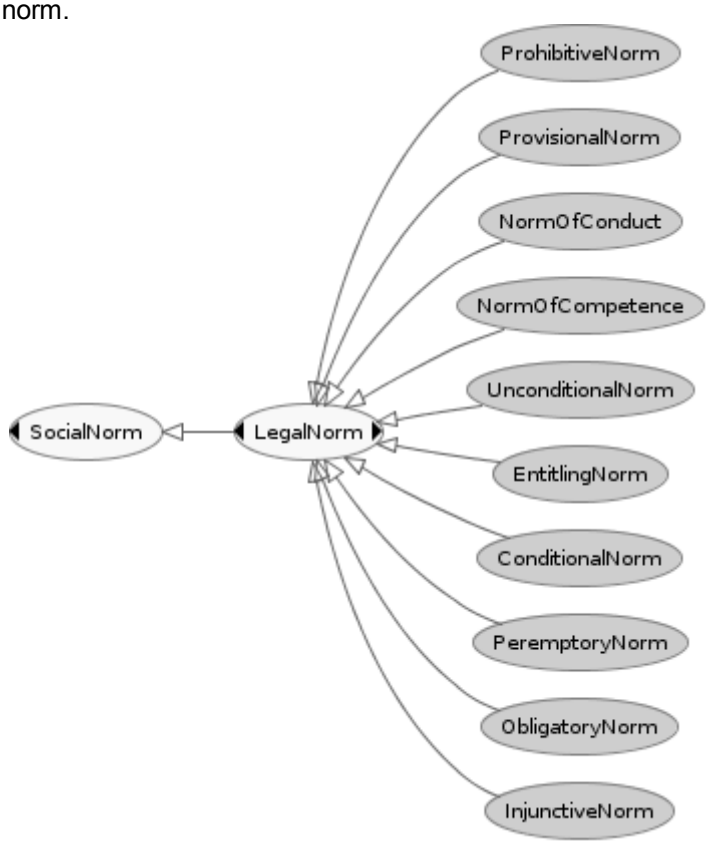


Fig. 4. The types of legal norms.

Although there are different views on what constitutes a legal norm (*LegalNorm)*, for the purpose of this ontology it is defined as a social norm that is sanctioned by the state. It is a central class of the model. A legal norm is a rule of conduct in a society that contains a rule on the application of a sanction in the case of its violation. Legal norms describe and prescribe (dis)allowed legal relations. Since the state of the legal system is a set of legal states of legal subjects (the set of their rights and duties), they also describe

and prescribe (dis)allowed states of the legal system. The legal norm and its different types are shown in Figure 4.

Legal norms can be classified according to legal relations they regulate and elements they contain. Prohibitive norms (*ProhibitiveNorm*) regulate legal relations that contain prohibitions. Provisional norms (*ProvisionalNorm*) contain dispositive disposition. Norms of conduct (*NormOfConduct*) regulate legal relations that contain right or duty (or equivalently contain categorical, alternative or dispositive dispositions). Norms of competence (*NormOfCompetence*) regulate legal relations that contain competence (or equivalently contain discretionary disposition). Unconditional norms (*UnconditionalNorm*) do not contain disposition hypothesis. Entitling norms (*EntitlingNorm*) regulate legal relations that contain a right. Conditional norms (*ConditionalNorm*) contain disposition hypothesis. Peremptory norms (*PeremptoryNorm*) contain imperative disposition. Obligatory norms (*ObligatoryNorm*) regulate legal relations that contain obligations. Injunctive norms (*InjunctiveNorm*) regulate legal relations that contain a duty. Those classes of legal norms are not necessarily mutually exclusive. For example, the definition of norm of competence is shown in Listing 1.

```
NormOfCompetence
  a owl:Class;
  owl:equivalentClass [
    a owl:Class;
    owl:intersectionOf (
      LegalNorm [
        a owl:Restriction;
        owl:onProperty hasElement;
        owl:someValuesFrom DiscretionaryDisposition
    ])
  ];
  owl:equivalentClass [
    a owl:Class;
    owl:intersectionOf (
      LegalNorm [
        a owl:Restriction;
        owl:onProperty regulates;
        owl:someValuesFrom [
          a owl:Restriction;
          owl:onProperty hasElement;
          owl:someValuesFrom Competence
        ]
    ])
  ].
```

Listing 1. The definition of the norm of competence.

Unlike some of the reviewed ontologies, our ontology does not specify the concept of constitutive norm, since that concept is not in the focus of the paper.

A legal norm is a basic building block of the legal system that is being modeled. It is used for modeling (a part of) a legal system that is going to be transformed into legislation that formulates it. It is also used for retrieval of legal norms and legislation using its properties. A legal norm has one or more

elements, regulates one or more legal relations, implements one or more policies, is a part of a legal institution, is contained in legislation, enters into force, is repealed and has efficacy on particular dates. Legal norm's properties are shown in Figure 5.

Fig. 5. The legal norm's properties.

Each legal norm consists of two main elements: a disposition and a sanction. A disposition (*Disposition*) is a rule of conduct in a society. A sanction (*Sanction*) is a rule of conduct of both the subject that has violated the disposition and the state (organization) that is mandated to use the appropriate measure on the violator. The subsidiary elements of legal norms are a disposition hypothesis, a sanction hypothesis and an exception. A disposition hypothesis (*DispositionHypothesis*) is the condition under which a subject has a right or a duty to act according to the disposition. A sanction hypothesis (*SanctionHypothesis*) is the condition of the application of the sanction. Violation of the disposition (a legal offense) is the necessary condition for the application of the sanction, but not the sufficient condition since further conditions may apply. Exception (*Exception*) limits the applicability of a norm.

There are several classes of dispositions. A categorical disposition (*CategorialDisposition*) is a disposition that describes and prescribes one and only one conduct. An alternative disposition (*AlternativeDisposition*) is a disposition that describes and prescribes one conduct from a set of alternative conducts that a subject can choose. A discretionary disposition (*DiscretionaryDisposition*) is a disposition that empowers a subject to regulate behavior of other subjects. Those classes of dispositions are the subclasses of imperative disposition (*ImperativeDisposition*). A dispositive disposition (*DispositiveDisposition*) is a disposition that describes and prescribes a conduct, but empowers a subject to create another disposition instead. The subject has to comply with the rule of conduct only if he/she does not create another disposition.

Sanctions can also be classified into imperative sanctions (that can further be classified into categorical sanctions, alternative sanctions and discretionary sanctions) and dispositive sanctions, although categorical sanctions are almost exclusively used. The norm element and its subclasses are shown in Figure 6.

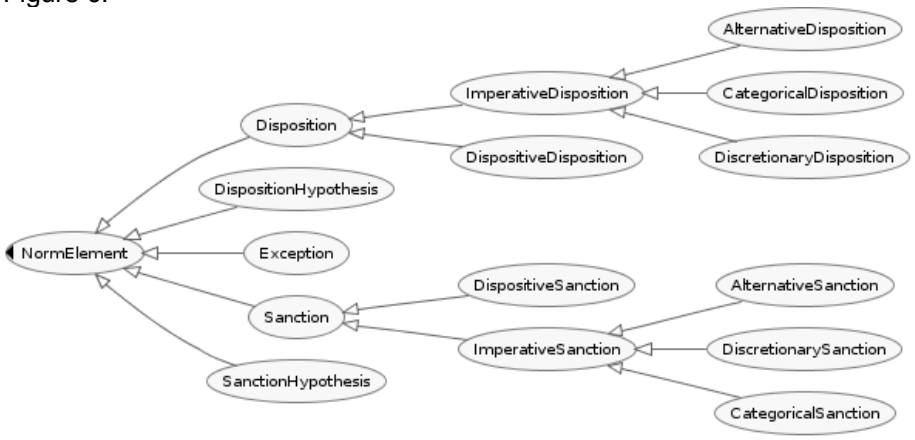


Fig. 6. The legal norm element and its subclasses.

Legal norms do not have textual formulation. Its elements have it. Legal norms are not directly connected with their textual formulations since different elements of legal norms can be contained in different (parts of) legislation. The element of legal norm can be formulated as a plain text or an URI reference to the XML element that formulates the norm element.

The element of a legal norm is used in several ways. Firstly, it is used to connect the content (legal norms) and the form of legislation (its text). Secondly, it is used for browsing legal norms by their elements since different legal norms can share same elements (e.g. different norms can have same sanction, the sanction or the disposition hypothesis of one legal norm can be the disposition of another, etc.).

A legal system (*LegalSystem*) is a set of legal norms arranged in a series of units that are connected with each other in a non-contradictory whole. Those units are a legal norm (*LegalNorm*), a legal institution (*LegalInstitution*), a legal branch (*LegalBranch*) and a legal area (*LegalArea*). A legal institution is a set of legal norms that regulate the same legal relation (or few similar legal relations) with the same policy (e.g. ownership, marriage, privacy, etc.). It should not be confused with an (state) organization although these concepts are related since (state) organizations are created in order to apply legal norms. A legal branch is a set of legal institutions (e.g. civil law, criminal law, family law, etc.). A legal area is a set of legal branches (e.g. public law, private law, national law, international law, etc.).

The purpose of those concepts is to organize legal norms into legal system in an explicit manner. This is made possible by having a property (*isPart*) that

specifies that an individual belonging to one unit is a part of an individual belonging to another unit. The classification elements are shown in Figure 7.

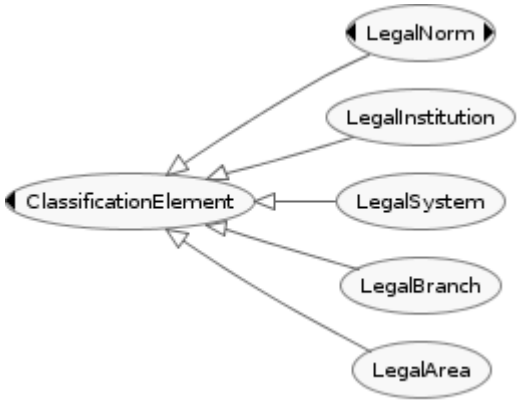


Fig. 7. The classification elements.

Legal norms are also implicitly organized with relations expressed by the Latin phrases *lex posterior derogat legi priori* (more recent law prevails over an inconsistent earlier law), *lex superior derogate legi inferiori* (a superior law prevails over an inconsistent inferior law) and *lex specialis derogat legi generali* (a specific law prevails over an inconsistent general law) that can be inferred from the model. The first relation can be inferred from the dates on which norms entered into force. The second relation can be inferred from the hierarchical relations between legal subjects that enacted legislation that contains norms. The third relation can be inferred from hierarchical relations between legal relations that are regulated by norms.

The structure of the legal system is used for retrieval of legal norms and legislation that formulates it.

A legal fact (*LegalFact*) is a fact that influences creation, modification or termination of legal relations (rights and duties). In other words, it is a fact that has legal consequences. It is usually described by disposition and sanction hypotheses.

An act (*Act*) is a change of state of things that is influenced by an agent (an agent is a subject that acts).

A mental act (*MentalAct*) is a change of mental state of a subject. This change is always influenced by an agent (subject itself), so it is an act.

The term legal act (*LegalAct*) has two main connotations. The first connotation of this term (its content, its subject matter) is a mental act that has legal consequences, that changes state of the legal system by changing legal states of legal subjects (their rights and duties). It has two parts. The main part of its content is a statement of will that has legal consequences. The subsidiary part of its content is the naming of the act itself (usually consisting of the type of the act, the subject that enacted it, legal grounds for its enactment, the place and the time of enactment, the procedure by which it was

enacted, the goal for its enactment, etc.). It is represented with an URI (e.g. accordance with URN:LEX [21] specification).

The second connotation of this term (its form) is the materialization of mental act that has legal consequences, usually expressed by natural language. The form of a legal act is a set of material means with which it is created and expressed. The legal theory distinguishes three main elements of its form: the subject, the procedure and the materialization. The subject is the body that is authorized to enact a legal act. The procedure is what is needed for its enactment. The materialization is the accommodation to sensory perception and expression of the legal act. The legal act and its subclasses are shown in Figure 8.

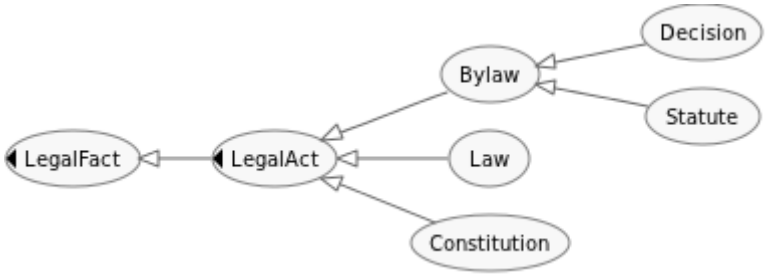


Fig. 8. The different types of legal acts.

Information about the subject, the procedure and the materialization is expressed by properties shown in Figure 9.

Fig. 9. The legal act's properties.

Those properties are used to retrieve legislation and legal norms contained in it (since legal act is explicitly connected with legal norms it contains).

It should be noted that the legal act in not a common term in the English language and countries with the common law legal system in general. For the purpose of this paper, the second connotation of this term, when contains (mostly) abstract and general legal norms, is synonymous with the term legislation.

4. An Example of the Model's Application

The Law on Personal Data Protection [3] of the Republic of Serbia regulates acquisition and processing of data within the context of protecting privacy of individuals.

We have represented both the form and the content of this law. The content of this law (i.e. norms it contains) is represented in RDF according to the OWL model described in Section 3. The modeling procedure is as follows: determine the scope, determine the policy, model social relations in the scope (and its elements), model legal norms that regulate those relations according to the chosen policy (and its elements) and express elements of those norms as plain text or XML. Due to space constraints, the original model of the law expressed in N3 notation is available at [20].

As noted, this system of legal norms is inferred from existing legislation. The procedure could also be reversed. Legislation could be textually formalized starting from the system of legal norms.

The form of this law (textual formulation of norms) is represented using the CEN MetaLex compatible model of legislation similar to the model described in [2].

The CEN MetaLex is intended to impose a standard view of legislation in order to facilitate information exchange and software interoperability. To meet those requirements, the CEN MetaLex defines mechanisms for XML schema extension, addition and extraction of metadata and implementation of identification mechanisms.

The CEN MetaLex schema defines abstract, generic and concrete types and declares abstract and generic elements. Abstract data types correspond to legal documents design patterns [21]. To enable the use of substitution groups in the declaration of conforming elements, the abstract types have corresponding elements. The CEN MetaLex schema contains generic types for each abstract type. Generic elements are declared for each generic type. They may be instantiated. Concrete types are included for all abstract types. They should be used for defining subtypes or elements conforming to the specification. In order to be compliant with the CEN MetaLex specification, each declared element has to be of a concrete type and has to have one of the abstract elements as its substitution head.

Legislative drafting guidelines of the National Assembly of the Republic of Serbia are regulated by [22]. All legislation enacted by the National Assembly has to be written in accordance with those guidelines. According to [22], based on its form, legislation is structured into parts, chapters, sections, subsections, articles, items, points, subpoints and lines. The CEN MetaLex specification has been extended in order to comply with [22]. New elements were declared for each structural parts of legislation. A full XML schema along with several examples of the legislation represented according to this model is available at [20].

The CEN MetaLex metadata is represented by RDF statements (subject, predicate and object). An OWL schema that specifies the allowed values of subjects, predicates and objects has been developed. It defines general con-

cepts, concepts that identify the document and concepts that are citations of other documents [23]. Only the subset of metadata specified in [23] was used in the CEN MetaLex representation of legislation presented in this paper. That subset contains classes and properties that were necessary for naming of legislation. RDFa was used as a method of serialization of RDF triplets.

The CEN MetaLex specification does not define the syntax or the semantics of identifiers. It defines rules that naming conventions must satisfy in order to be compliant with the specification. The CEN MetaLex distinguishes identity of legislation at FRBR [24] work, expression, manifestation and item levels. Feature set has been chosen to identify uniquely legislation at work, expression and manifestation levels. Those features are serialized both into RDF metadata in conformance with [23] and into IRIs of the syntax in conformance with [25].

This representation is straightforward. Each formal element of the legislation (e.g. article, item, point, etc.) is represented by a corresponding XML element that has *id* attribute as a unique identifier. The XML element *provision* represents textual formulation of a part (element) of legal norm. The original document is available at [20].

It is important to notice that the RDF representation of the elements of legal norms (content of legislation) and the XML representation of provisions (form of legislation) are connected with *asURI* property. Therefore, legal norms are connected with their formulations (elements of legislation), while legislation is connected with its content (legal norms).

The duality between the form and the content of legislation was used as a basis for developing a prototype expert system for semantic browsing of legislation. It stores legislation as XML documents in accordance with the CEN MetaLex specification and legal norms as RDF triplets in accordance with the model described in this paper. The usage of this prototype is described in the remaining of this section.

User interface of the prototype consists of several tabs. Legislation is shown in *Content* tab. It contains several views that can be shown by pressing button ⩾ or hidden by pressing button ⩽. Furthermore, it is possible to show the table of content (*Table of Content* tab), the list of attachments (*Attachments* tab), the list of bylaws (*Bylaws* tab) and the legislation metadata (*Metadata* tab). The table of contents and the list of attachments are automatically generated from the XML representation of legislation. The list of bylaws and the legislation metadata are automatically generated from the RDF representation of norms contained in legislation.

Legislation can be browsed by form or by content. It is browsed by form simply by following textual hyperlinks between different elements of legislation (articles, items, points, etc.) or different legislation altogether.

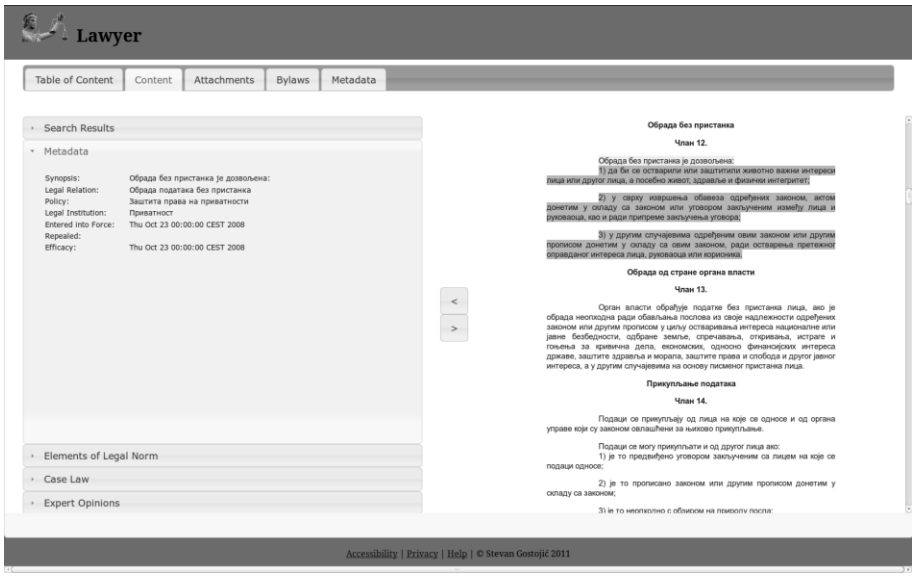


Fig. 10. Metadata view.

Fig. 11. Elements of Legal Norm view.

Browsing by content is facilitated in the following way. When a provision that formulates an element of a legal norm is clicked, provisions that formulate all the elements of that legal norm are shown in different colors - disposition is yellow, disposition hypothesis is lime, sanction is aqua, sanction hypothesis is fuchsia and exception is silver. The *Metadata* view displays norm metadata

(Figure 10). The *Elements of Legal Norm* view displays a list of elements of this norm (Figure 11). When an element of this norm is clicked, its textual formulation is shown in the *Content* tab. The *Search Results* view displays a list of legal norms that contains the elements formulated by the provision that was clicked (Figure 12). When a legal norm from the list of legal norms displayed in the *Search Results* view is clicked, it is displayed in the similar manner. The *Case Law* view displays a list of case laws that are related to the selected norms. Similarly, the *Expert Opinions* view displays a list of expert opinions related to the selected norms.

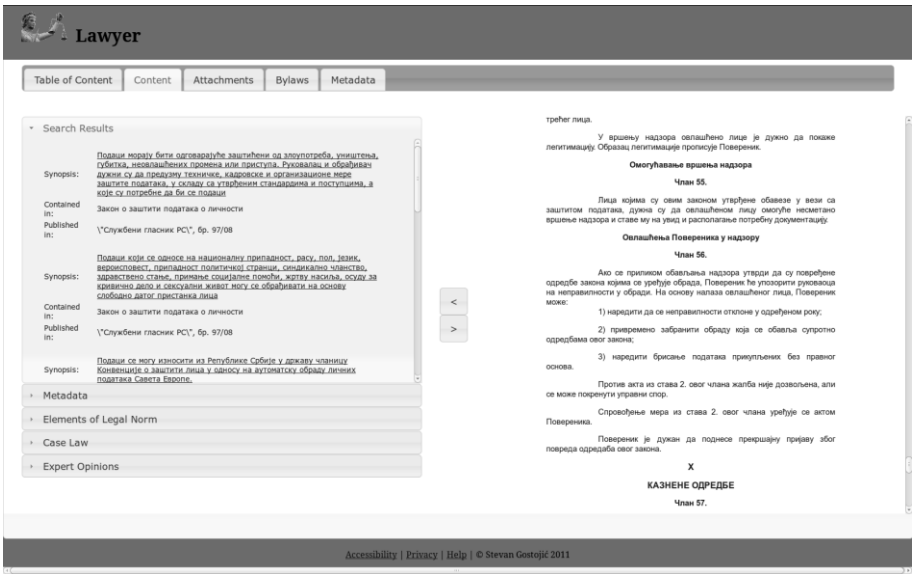


Fig. 12. Search Results view.

5. Conclusion

This article describes a formal model of legal norms developed using OWL. It is intended for semiautomatic drafting of legislation from a system of legal norms it contains and semantic retrieval and browsing of legislation annotated with the information about legal norms they contain. The model is verified by applying it to an existing piece of legislation and by developing a prototype application intended for semantic browsing of legislation that can solve the problem of legal rule fragmentation.

The main contribution of the paper is the adoption of the structural view of the legal system and subsequent definition of all relevant concepts of the model using the elements of the legal relation and the elements of the legal norm. While reviewed ontologies connect legal norm with the action or behavior of the legal subject it describes or prescribes, we connect it with legal rela-

tions they regulate. To the best of our knowledge, no other model of legal norms used this approach.

Nevertheless, modeling of a system of legal norms contained in legislation requires considerable time and expertise. Apart from being acquainted with OWL and the described model, a person responsible for this task is required to be an expert in normalized legal drafting as well as in the area that is being regulated. Therefore, our future work is directed in two complementary directions.

There are multiple research projects with the goal to develop legislative drafting environment [26], but the semiautomatic application of legislative drafting guidelines is on the rudimentary level. None of those tools supports semiautomatic drafting of legislation starting from its semantics. One possible solution to this problem is the use of a modeling tool that can generate draft legislation from the model of a system of legal norms. That way, apart from improving drafting process and the quality of resulting legislation, a model of a system of legal norms would be a byproduct of the drafting process. Drafting of legislation can be automated to some extent by transforming the model of a system of legal norms in accordance with the described formalism, using transformations described in specific legislative drafting guidelines, to the model of legislation in accordance with the CEN MetaLex specification (cf. [1]). Although this process cannot be completely automatic, the structure of the draft legislation can be a considerable help to the legislative drafter and annotated legislation can be used for semantic retrieval and browsing.

Retrieval and browsing of legislation can be facilitated by exploiting duality of legislation and legal norms and the structure of the legal relation, the legal norm and the legal system. Developing a prototype expert system for semantic retrieval of legislation is a natural continuation of the research on browsing of legislation. Semantic retirieval is based on the meaning of legislation (the legal norms contained in it).

Furthermore, the model could be expanded to include specific and concrete legal norms, although that can effect computing properties of the model since the expanded model would not necessary be the OWL DL model. Ontology presented in this paper can be integrated with existing (legal) ontologies, although this was not the focus of the research described in this paper.

Acknowledgments. The research presented in this paper was financed by the Ministry of Education and Science of the Republic of Serbia as part of the research project “Intelligent Systems for Software Product Development and Business Support based on Models” (grant no. 44010). The authors would also like to thank anonymous reviewers for suggestions that considerably improved the quality of the paper.

References

1. Biagioli, C., Cappelli, A., Francesconi, E. and Turchi, F.: Law Making Environment: Perspectives. In: Biagioli, C., Francesconi, E. and Sartor, G. (eds.): Proceedings

of the V Legislative XML Workshop, European Press Academic Publishing, Florence, 267–283. (2007)
2. Gostojić, S., Milosavljević, B. and Konjović, Z.: Modeling MetaLex/CEN Compliant Legal Acts. In: Szakal, A. (ed.): Proceedings of the 8th International Symposium on Intelligent Systems and Informatics, IEEE, New York, 285–290. (2010)
3. Law on Personal Data Protection ("Službeni glasnik RS", br. 97/2008, 104/2009). (in Serbian)
4. Biagioli, C. and Grossi, D.: Formal Aspects of Legislative Meta-drafting. In: Francesconi, E., Sartor, G. and Tiscornia, D. (eds.): Proceeding of the 2008 Conference on Legal Knowledge and Information Systems: JURIX 2008: The Twenty-First Annual Conference, IOS Press, Amsterdam, The Netherlands, 192–201. (2008)
5. Sartor, G.: A Formal Model of Legal Argumentation. Ratio Juris, Vol. 7 No. 2, 177–211. (1994)
6. Gordon, T.: The Legal Knowledge Interchange Format (LKIF). University of Amsterdam, Amsterdam, The Netherlands. (2008)
7. Kralingen, R.: A Conceptual Frame-based Ontology for the Law. In Proceedings of the First International Workshop on Legal Ontologies, University of Melbourne, Melbourne, 15-22. (1997)
8. Breuker, J. and Hoekstra, R.: Epistemology and Ontology in Core Ontologies: FOLaw and LRI-Core, Two Core Ontologies for Law. In: Gangemi, A. and Borgo, S. (eds.): Proceedings of the Workshop on Core Ontologies in Ontology Engineering, RWTH, Aachen, 15-27. (2004)
9. Gangemi, A.: Design Patterns for Legal Ontology Construction. In: Casanovas, P., Biasiotti, M., Francesconi, E. and Sagri, M. (eds.): Proceedings of the 2nd Workshop on Legal Ontologies and Artificial Intelligence Techniques, RWTH, Aachen, 65-85. (2008)
10. Rubino, R., Rotolo, A. and Sartor, G.: An OWL Ontology of Fundamental Legal Concepts. In: van Engers, T. (ed.): Proceeding of the 2006 Conference on Legal Knowledge and Information Systems: JURIX 2006: The Nineteenth Annual Conference, IOS Press, Amsterdam, 101–110. (2006)
11. Breuker, J., Hoekstra, R., Boer, A., van den Berg, K., Sartor, G., Rubino, R., Wyner, A., Bench-Capon, T. and Palmirani, M.: Deliverable 1.4: OWL Ontology of Basic Legal Concepts (LKIF-Core). University of Amsterdam, Amsterdam, The Netherlands. (2007)
12. Valente, A., Breuker, J., Brouwer, B.: Legal Modeling and Automated Reasoning with ON-LINE. International Journal of Human-Computer Studies, Vol. 51, No. 6, 1079–1125 (1999)
13. http://www.estrellaproject.org (current 1 June 2011)
14. Olbrich, S. and Simon, C.: Process Modelling Towards e-Government – Visualisation and Semantic Modelling of Legal Regulations as Executable Process Sets. Electronic Journal of e-Government, Vol. 6 No. 1, 43–54. (2008)
15. Lukić, R. and Košutić, B.: Introduction to Law, IRO Naučna knjiga, Belgrade, Serbia. (1988) (in Serbian)
16. Vukadinović, G.: Theory of State and Law, Futura publikacije, Novi Sad, Serbia. (2006) (in Serbian)
17. Pajvančić, M.: Legislative Drafting, Advokatska komora Vojvodine, Novi Sad, Serbia. (1995) (in Serbian)
18. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A. and Schneider, L.: Deliverable 1.7: The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. ISTC-CNR, Padova, Italy (2002)

19. Gruber, T. Toward Principles for the Design of Ontologies Used for Knowledge Sharing: International Journal of Human-Computer Studies, Vol. 43 No. 5–6, 907–928 (1995)
20. http://informatika.ftn.uns.ac.rs/legal (current 1 June 2011)
21. Vitali, F., Di Iorio, A. and Gubellini, D.: Design Patterns for Descriptive Document Substructures. Paper presented at The 2005 Extreme Markup Languages Conference, 1–5 August 2005, Montréal. (2005)
22. Legislative Drafting Guidelines ("Službeni glasnik RS", br. 21/2010). (in Serbian)
23. http://svn.metalex.eu/svn/MetaLexWS/branches/latest/metalex-cen.owl (current 1 June 2011).
24. International Federation of Library Associations and Institutions: Functional Requirements for Bibliographic Records, International Federation of Library Associations and Institutions, The Hague, The Netherlands (2007). Available: http://www.ifla.org/en/publications/functional-requirements-for-bibliographic-records (current 1 June 2011)
25. Spinosa, P., Francesconi, E. and Lupo, C.: A uniform resource name (URN) namespace for sources of law (LEX). Internet Engineering Task Force, Fremont (2011)
26. Arsovksi, S., Konjovic, Z., Milosavljevic, B., Gostojic, S.: „Legislative editors based on open standards and open source. YUINFO 2010, Kopaonik, Serbia (2010)

Stevan Gostojić has a Ph.D. in electrical engineering and computer science from University of Novi Sad. Currently, he works as an assistant professor at Faculty of Technical Sciences in Novi Sad. His research interests are legal informatics, e-government, document management, business process management, distributed computing, WWW, XML and semantic web.

Branko Milosavljević is an Associate Professor in the Faculty of Technical Sciences, University of Novi Sad, Serbia, where he earned his doctoral degree in Computer Science. His research interests include information retrieval, document management, access control, and digital libraries.

Zora Konjović is Full Professor in the Faculty of Technical Sciences, Novi Sad, Serbia. Dr. Konjović received her Bachelor degree in Mathematics from the University of Novi Sad, Faculty of Science and Master degree and Ph. D. degree both in Robotics from the University of Novi Sad, Faculty of Technical Sciences. She has participated in six scientific and more than thirty professional projects; she was the project leader for five of these. Dr. Konjović has published more than 180 scientific and professional papers.

Received: August 04, 2011; Accepted: July 02, 2012.

DOI:10.2298/CSIS111127040L

Indexing moving objects: A real time approach[1]

George Lagogiannis[1], Nikos Lorentzos[1], and Alexander B. Sideridis[1]

[1]*Agricultural* University of Athens, Iera Odos 75, 11855 Athens, Greece
{lagogian, lorentzos, as}@aua.gr

Abstract. Indexing moving objects usually involves a great amount of updates, caused by objects reporting their current position. In order to keep the present and past positions of the objects in secondary memory, each update introduces an I/O and this process is sometimes creating a bottleneck. In this paper we deal with the problem of minimizing the number of I/Os in such a way that queries concerning the present and past positions of the objects can be answered efficiently. In particular we propose two new approaches that achieve an asymptotically optimal number of I/Os for performing the necessary updates. The approaches are based on the assumption that the primary memory suffices for storing the current positions of the objects.

Keywords: Persistence, I/O complexity, Indexing structures.

1. Introduction

Objects that change their position and/or shapes over time introduce large spatio-temporal data sets. The efficient manipulation of such data sets is crucial for an increasing number of computer applications (location aware services, traffic monitoring etc). Considering in particular, moving objects as vehicles that move in a city, we can think of many interesting queries such as "find the closest police car", or "find the number of vehicles that went through the center of the city between 11:00 and 13:00".

In the real time version of our spatiotemporal problem, we can consider a client-server architecture where each moving client (object) sends its position to the server, at discrete times. The server collects information reports (messages) of the form (object Id, current-cell, current time) from the moving objects, every *P* seconds. We assume that the objects move in a 2 d space.
Queries on such data sets may be of a "historical" kind or may be posed strictly on the *current* position of the objects. This paper does not deal with the present time. By *current position* of an object we mean the position indicated by the last message sent by the object. The actual current position is unknown, and one can only guess, according to the latest position, and the speed vector of the object. Thus we only deal with the past. The term "real

[1]An abstract version of this work was presented in WSKS 2010 (see [8])

time", is used to denote that the updates on the data structures (in secondary memory) caused by the messages (sent by the vehicles) are not postponed, but instead, these structures are updated with every incoming message.

In building a system to index moving objects, we have two alternatives for indexing the space involved (a 2-d space in our case), *static* and *dynamic* indexing. In static indexing, the 2-d space is divided into "cells" and the area occupied by each such cell does not change during the monitoring period, i.e., it is static. In dynamic indexing, the 2-d space is divided into regions in such a way that, at all times, there is a minimum number of objects in every region. To satisfy this property, we need to update the regions as objects move (hence the regions are dynamic). In this paper we provide solutions based on both static and dynamic indexing strategies, aiming at minimizing the number of I/Os needed to store the messages sent by the objects. Assuming that with each I/O we can store B such messages (i.e. B messages fit into a disk block), we conclude that the minimum number of I/Os can be achieved if we manage to store B messages with each I/O. If we manage to store $c*B$ messages per I/O, (where c is a constant less than 1) then we say that our solution is *asymptotically* optimal. Such solutions are present in this paper.

Optimizing the I/Os of existing multidimensional indexing structures (mainly the R-tree) is the target of many recent efforts (see [3], [4], [6], [11], [12], [14]). A common part of most of these solutions is a secondary index structure, used for accessing the leaf of the main indexing structure that contains a given object. This secondary index structure is used to avoid the multiple paths search operation in the R-tree during the top-down update. This way a bottom-up approach is proposed.

Compared with the related work described above, our work differs because of the combination of the following three characteristics:

i) We use a worst case efficient data structure instead of an R-tree, since the R-tree is not very efficient under a large amount of updates. The worst case framework which we apply is important for real time applications, where the data structures involved should be completely predictable with respect to their time complexities. Searching for example for the closest taxi requires a predictable amount of time because the positions of the objects change rapidly and the taxi closest, one minute ago, may not currently be the closest taxi. Tight bounds tend to make such applications more reliable and, in this sense, reliability is really promoted by using a worst case efficient indexing structure and in particular, partially persistent B-trees (see [2], [13]).

ii) We aim at storing not only the present positions of the objects, but also the past ones. The past positions are crucial for the answering of historical queries and such queries are certainly of interest.

iii) In contrast with most of the related work, which is implementation oriented, a practical implementation is not our objective, i.e. our work should not be seen as competitive to practical, implementation-oriented solutions. It must be noted that in reality, a simple observation of the way by which the objects tend to move may prove to be more useful than the theoretical asymptotical optimality, which we provide. For example, one could logically neglect messages from motionless vehicles and, therefore, reduce the

number of I/Os. We do not make any assumptions or real life observations relevant to the way of the movement of the objects. Thus, our work should be seen as an approach into which many observations from real applications can be incorporated, in order for practical implementations to be created.

2. Problem definition

As is obvious, storing the past positions of objects requires the use of secondary storage because of the huge amount of data involved. Given that a large number of objects is being tracked, the main concern is to face the bottlenecks caused by the large volume of I/Os for storing into secondary memory the messages sent by the objects. The parameters involved are summarized as follows.

N: The maximum number of tracked objects.
M: The amount of main memory used.
P: The time period of communication between the objects and the base station, measured in seconds.
R: The number of I/Os per second supported by the hard disk of our system.
B: The number of messages that fit into a disk block.
W: The total number of messages received by the system during the tracking time.

An I/O may be of one of the following two types:

- *Message storing I/O*, which stores some (optimally O(*B*)) messages, into a disk block.
- *Rebalancing I/O:* This I/O is caused by the indexing structure.

The first type of I/O is caused by the incoming messages. For an example of an I/O of the second type, consider a message-storing I/O that inserts a new record into a leaf of a B-tree. This may cause a split of the leaf and of some of the ancestors of this leaf. The additional I/Os, required to rebalance the B-tree, are the rebalancing I/Os.

Since at most *B* messages can be stored into secondary memory by one I/O, it follows that the minimum number of I/Os that can be achieved is *W*/*B*. In fact, this number can be achieved by the following trivial solution: Each new message sent by an object is copied into a buffer, whose capacity is equal to *B* messages. When this buffer is filled, we store its *B* messages at the end of a secondary memory file and the buffer is then freed. Clearly, this solution achieves *W*/*B* I/Os all of which are message-storing, since no indexing structure is used.

Such a solution, though optimal with respect to the number of I/Os, does not efficiently answer queries concerning the objects, due to the lack of an indexing structure. Our objective is to achieve asymptotically optimal solutions with respect to the number of I/Os, that are still query efficient. In particular, we allow for the number of I/Os to be O(*W*/*B*) (i.e. the number of I/Os is multiplied by a constant factor) rather than *W*/*B* (of the trivial solution) and

show that this sacrifice is enough to achieve query efficient solutions. In conclusion, the solutions we present have the following property.

Property 1: To store the total number of messages (*W*) received by the system, *t*he required number of message-storing I/Os is O(*W*/*B*).

For our purposes, it is assumed that the primary memory is sufficiently large to store the current position of tracked objects. Assume, for example, that 5 million moving vehicles are being tracked in a city. Assume also that, for each vehicle, a tuple of *c* bytes is maintained in primary memory, containing the Vehicle Id and the necessary additional data. Then the primary storage required is not more than 5**c* Mbytes. Assuming that we use sophisticated data structures, this number has to be multiplied by only a small constant. Such an amount of primary memory is not considered to be prohibitive nowadays neither from a technical nor from an economical point of view.

As mentioned in the introduction, one can index the 2-d space where the objects move, statically and dynamically. In this paper we follow both approaches.

Our static indexing approach is based on a grid. We assign an indexing structure to each cell of the grid. Such an indexing strategy is suitable for the processing of range queries. In this paper we explore this strategy for the processing of spatiotemporal range predicates. A spatiotemporal range predicate is a pair (*S*, *T*) where *S* is a spatial constraint and *T* is a temporal constraint which can be either a time instance or a time interval. The output of the query is either the set of objects inside S at the time instance *T*, or the set of objects inside *S* at some time instance during the time interval *T*.

By indexing the space in a dynamic way, we are able to efficiently process spatio-temporal *nearest (k-nearest) neighbour* predicates. Such a predicate is a pair (*Q*, *T*), where *Q* is a point on the map and *T* can again be either a time instance or a time interval. The output is the nearest object (or the k-nearest objects) to *Q*, at the time instance *T* or during the time interval *T*.

The time complexities of the solutions provided are derived in the external-memory model of computation given in [1], i.e. we neglect the time spent for primary memory actions and the only measurement of efficiency we care about is the number of I/Os.

The remainder of the paper can be summarized as follows: The partially persistent B-tree, briefly presented in Section 3, represents the base structure for the description of the proposed approaches. Sections 4 and 5 aim at reducing the message storing I/Os. The approach in Section 4 is based on the static indexing of the involved space whereas in Section 5 dynamic indexing is discussed. In Section 6 we discuss rebalancing I/Os. In Section 7, we finally draw conclusions and discuss issues of further research.

3. Partial Persistence

Traditional data structures are *ephemeral*, in the sense that we do not maintain older versions, we only update the current version. Maintenance of the old versions leads to persistent data structures. There are three kinds of persistence: In *partial persistence*, the latest version can be updated, and the old versions can only be searched. In *full persistence*, all versions can be searched and be updated. Finally, in *confluent persistence*, one property is added, that two different versions can be merged.

In the seminal paper by Driscoll, Sarnak, Sleator and Tarjan, [5], two general methods are presented, that transform an ephemeral data structure into a partially persistent: The *fat-node* method (also achieving full persistence) and the *node-copying* method. By applying the fat-node or the node-copying method to an ephemeral (initial) structure we can create its partially persistent version.

A fat node corresponds to a node of the (initial) ephemeral structure. It can become arbitrarily big, and it contains the entire history of the corresponding ephemeral node. The node-copying method produces fixed-size nodes and it is optimal, i.e., the time complexity of the produced partially persistent structure is asymptotically equal to the time complexity of the (initial) ephemeral structure.

Applying the methods of [5] in secondary memory turned out to be a separate research area, because its straightforward application leads to a huge amount of wasted space. Having been inspired by the fat-node method, Lanka and Mays [10], proposed a method, called *fat field*, that reduces the space requirements of their data structure. In this method, the empty fields of a block in a fat node are used to store modifications of data fields, as long as they do not cause overflows. Using this method, they presented fully persistent B-trees which can also be used for the partially persistent case except that the time complexities achieved this way, are not optimal.

To achieve optimal partially persistent B+-trees, one must adjust the node-copying method to secondary memory. Such partially persistent B-trees have also been developed, in particular the Multi Version B-Tree (MVBT) by Becker et al. [2] and the Multi Version Access Structure (MVAS) by Varman and Verma [13]. These methods essentially share the same ideas. The approaches presented in the next sections are based on the partially persistent B-tree ([2], [13]). A brief description of these structures follows.

In general, a partially persistent B-tree is a modified B+-tree. Its internal nodes contain index records and its leaves contain data records. A data record contains the fields *key, start* (the time instance that the record was inserted into the tree*), end (*the time instance when the record was “deleted”), and *info* (information associated with the *key*). An index record contains the fields *key*, *start*, *end* and *ptr*, where *ptr* is a pointer to a node of the next level. The node pointed by the *ptr* pointer contains keys no less than *key*, has been created at the time instance *start* and has been copied at the time instance *end*. A data record is *active* (*live*) if its *end* field has value ‘$’, i.e. it has not been updated, “deleted” or copied to another node. If this is not the case, the

data record is *inactive* (*dead*). Thus, to "delete" a record we just set its *end* value to the current time. An index record is active if it points to an active node at the immediately lower level.

From the above description it follows that the current version of a partially persistent B-tree contains all the active data records. A node that contains active records is also called *active* otherwise it is *inactive*. Thus, the current version of a partially persistent B-tree contains all its active nodes. A node becomes inactive when it is rebalanced.

Figure 1 shows a possible instance of a partially persistent B-tree, and a simple scenario. At time 5 (upper part of the figure), the tree consists of two nodes, the root and one leaf, which contains all the data records. The figure shows that key A was inserted at time 1, key C was inserted at time 2 and was subsequently modified at times 3 and 4, and key F was inserted at time 5. Then, at time 6, key D is to be inserted. This insertion causes an overflow of the single leaf. Two new leaves are then created and the old leaf becomes inactive (all the inactive records appear shaded). The index record of the root, which points to the inactive leaf, also becomes inactive (shaded). The set of live records of the old leaf is sorted by key, is divided into two halves and each of these halves is copied to one of the two new leaves. Two new index records are created in the root. Their start value is the time at which the pointed leaves were created, i.e. time 6. To delete a record, we set its end-value to the current time instance and then count the remaining live records of the leaf. If they are too few, we may borrow some live records from a neighbour leaf, and create one or two new leaves.

The fact that one or two leaves may be created requires some brief discussion. In the scenario of Figure 1, the rebalanced leaf contains less than 5 active records. Since however 5 records fit into one leaf, one would expect that only one new leaf is needed. Instead, one can see that we have created two new leaves. Appropriate explanation for this decision is now justified by the following: In general, the number of new leaves created is dictated by our need to create "stable" leaves that will not be rebalanced soon. As an example, let us set to B the capacity of each node of the persistent tree. If the leaf being rebalanced contains B active records, then we can move all the active records to a new leaf but this leaf will immediately be rebalanced if a new insertion occurs inside it. The general rule is that we create one or two "stable" leaves, each of them requiring $\Theta(B)$ updates (insertions or deletions) in order to be rebalanced again. After a deletion, we count the number of active records inside the node containing the "deleted" record. If the number is smaller than a threshold, then we may transfer some active records to that node, from a neighboring node, or merge this node with a neighboring node, and create one or two stable nodes. It is easy to see that creating stable nodes is not a difficult task (further details can be found in [2] and [13]). Thus, partially persistent B-trees have the following property, which is important for our solutions.

Property 2: When a new node is created, it is able to tolerate $\Theta(B)$ updates until it becomes inactive.

In order to achieve Property 2, the minimum number of active records inside a node must be Θ(*B*) otherwise, a node will have to be merged before it "experiences" Θ(*B*) "delete" operations. Assuming for example that the minimum number is 4 then, by merging two nodes, we create one node with at least 8 active records. This node can then experience 4 deletion operations before it has to be merged again. The fact that the minimum number of records is set to Θ(*B*), leads to Property 3, which is also important for our solutions.

Property 3: If we navigate into the persistent structure at time instance *t*, each node we access has Θ(*B*) records, valid at this instance.

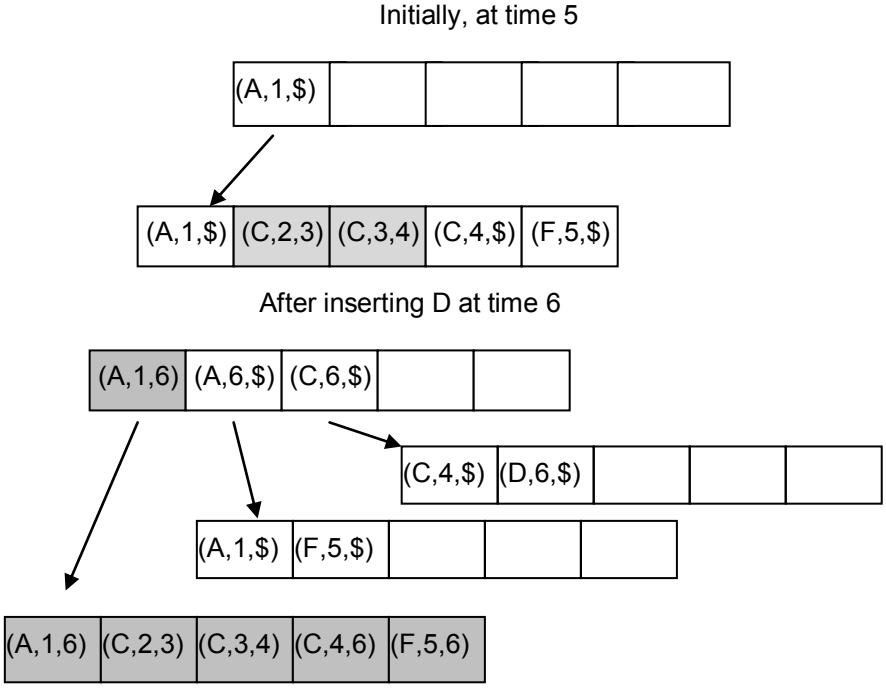


Fig. 1. A simple scenario of a partially persistent B-tree. The ptr-field for each record is visualized through an arrow pointing to the level below. The fields key, *start* and *end*, are shown in the same order. The shaded records are inactive and the remainder are active.

Let us now briefly describe the navigation inside a partially persistent B-tree. To search for a key *K* that is valid at time *t*, we start from the root. We ignore the records with *start* values greater than *t* and the records with *end* values less than *t*. From the remaining records we choose the one with greatest key value, less than or equal to *K*; if there are several such records, we choose that one with the greatest *start* value. We follow the pointer to the next level and we then apply the same procedure at this level. Note that this process introduces a unique path towards the leaf level, where a leaf is

accessed. The record will be found in the leaf if it really exists; otherwise, there was no record with key K valid at time t. Searching for example for element F at time instance 5 of Figure 1, we shall ignore records (A, 6, $) and (C, 6, $) because they were created after time instance 5. If, on the other hand, we search for element D at time 6, we shall ignore record (A, 1, 6) in the root of the structure, because this record was "deleted" at time 6. From the remaining records, we choose (C, 6, $) and we then follow the pointer indicated by this record.

The space consumption of optimal partially persistent B-trees is $O(m/B)$ blocks (where m is the total number of updates) and updates can be performed in a $O(\log_B(m/B))$ worst case time. In the amortized case, the update time is constant (see [2], [13]).

4. Static Indexing

We consider a grid on our 2-d map. In the static indexing we assume that the horizontal and vertical lines of the grid are determined in advance and they do not move during the monitoring period. Hence, the 2-d map is divided into static cells. Each incoming message is a tuple (O_{id}, C_{id}, t), where O_{id} is the id of the object that sent the message, C_{id} is the id of the cell that contains O_{id}, and t is the time at which the message was sent. For simplicity, we assume that each object can determine its current cell, i.e., it has some computational power. If this is not the case, the current cell can easily be determined by the system, with a simple calculation. The objects inside each cell are indexed by a partially persistent B-tree. Each time an object leaves a cell C_2 and enters another cell C_1, its record, which is located in C_2 is set to inactive, by replacing the value $ of its *end* field by the time at which the object sent the message. Next, a new record for this object is inserted into the persistent B-tree of C_1. Its *start* value is set equal to the time at which the message was sent, and its *end* value is set to $. This approach is described in Subsections 4.1 and 4.2. To avoid complicated details, we assume that if an I/O is needed, it is performed immediately. Note that although this is an unrealistic assumption, it allows for simplifications. The realistic assumption is that if an I/O is needed, it may not be completed instantly, because another I/O is performed at the same time. Thus, the I/Os become *pending I/Os*, and are inserted into a list called *I/O-list*. Message storing I/Os are executed by extracting pending I/O requests from the I/O list in FIFO order. In Subsection 4.3, we analyze the approach by taking into account this last, realistic assumption. Finally, in Subsection 4.4, we explore the efficiency of the approach for the handling of spatiotemporal range predicates.

4.1 Data Structures

For each cell C_i of the grid, we maintain in secondary memory a partially persistent B-tree, called PBC_i. In primary memory we maintain the following data structures:

- For each cell C_i of the grid, we maintain an indexing structure called *active_PBCi*. Let C_i be a cell. PBC_i contains both active and inactive nodes. The *active_PBC*$_i$ is the tree defined by the active nodes of PBC_i and the pointers that connect these active nodes. For every leaf V of the *active_PBC*$_i$, there is a leaf X in PBC_i which satisfies the following property: *At the time V was created, X was also created to be identical to V*. We call *X, image of V*, and we store into V a pointer towards X.
- A table A containing the tracked objects. Suppose that we receive a message from object i. Then entry A[i] contains the current cell of the object.

4.2 Algorithm to Handle Incoming Messages

Suppose we receive a message (O_i, C_k, t). The algorithm for the processing of this message follows, and the result of the algorithm is visualized in Figure 2.

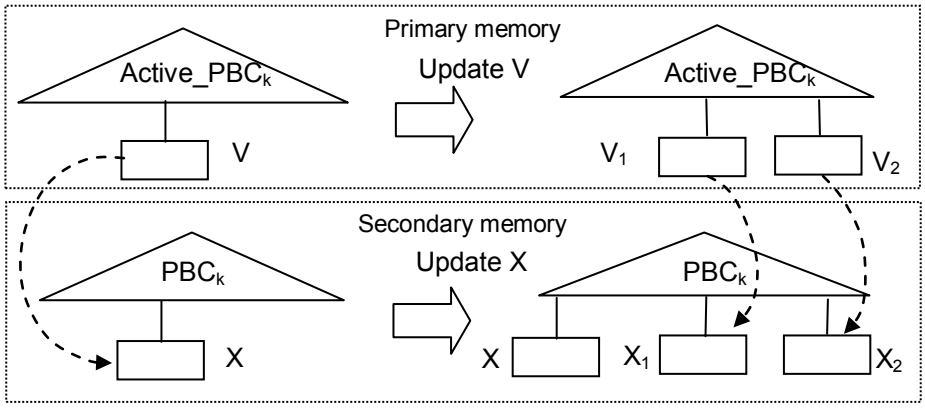


Fig. 2. Deactivating leaf V in cell C_k, leads to updating its image leaf X on the disk

Step 1: We go to A[i] and find the current cell of O_i. Let C_j be that cell. If C_k = C_j, we do nothing. Otherwise we store C_k, in A[i] and proceed to Step 2.

Step 2: We find the appropriate leaf V in the *active_PBC*$_k$ and insert into it the new tuple. If V is not full, we are done. If V is full we may have either to split, or merge V with a neighboring leaf. In either case, one or two new

leaves must be created. Figure 2 shows the actions of Step 2, in the case at which V is split into two leaves, named V_1 and V_2. We execute the insertion algorithm of the partially persistent B-tree on the $active\_PBC_k$, with one difference: we throw away all the inactive nodes. For example, in Figure 2, leaf V is thrown away when the algorithm finishes. By following the pointer from V we reach X, the image of V. We then update X, to be identical to V. Next, we proceed with the insertion algorithm on PBC_k, and we create the image leaf of each new leaf created by the insertion algorithm in the $active\_PBC_k$ (X_1 is the image leaf of V_1 and X_2 is the image leaf of V_2). We connect each new leaf in primary memory, with its image leaf in secondary memory.

Step 3: In the $active\_PBC_j$, we find the leaf that contains the tuple of O_i. We execute the deletion algorithm of the partially persistent B-tree on the $active\_PBC_j$, in order to delete the tuple of O_i, in the same way we executed the insertion algorithm in Step 2.

4.3 Analysis of the Solution

Lemma 1: If W is the total number of messages received by the system, then the approach of this section stores these messages in $O(W/B)$ message storing I/Os, i.e., Property 1 holds:

Proof: When a leaf in primary memory becomes inactive, all its records are stored in the image leaf. According to the algorithm of the partially persistent B-tree, every leaf of an active_PB remains active until $\Theta(B)$ insertions or "deletions" (i.e., record deactivations) occur (due to Property 2). Each insertion or "deletion" corresponds to a message that has not been stored. Thus, when a leaf is deactivated, $\Theta(B)$ *new* messages are stored in secondary memory. By *new*, we mean messages that have not been stored in secondary memory earlier. As a result, in order to store all the W messages, we need $O(W/B)$ I/Os. □

Lemma 2: Let S_1 be the amount of primary memory occupied by active nodes. Then, S_1 is $\Theta(N)$.

Proof: Let u be an active node of an active_PB. Then u contains active records. We know that, in every node, the total number of records is at most B and the number of active records is $\Theta(B)$ (due to Property 3). Since the number of active records stored in all the active leaves of active_PBs is N (equal to the number of tracked objects), we conclude that the total number of records stored in all the active leaves of active_PBs is $\Theta(N)$. Adding the space occupied by the internal nodes, the total space is multiplied by a constant less than 2. Therefore we conclude that the total space consumed by the active_PBs is $\Theta(N)$. □

At this point, we add to the approach already presented in Subsection 4.2, the realistic assumption that *an I/O list exists*. Whenever a leaf in main memory is deactivated, it is inserted into this I/O list. I/Os in this list are processed in a FIFO order. Once the I/O indicated by the leaf has been served, the space in main memory which was occupied by the leaf is set free.

Due to this, the definition of active_PBs has to be revised as follows: *If C_i is a cell of the grid then the active_PBCi contains all the active nodes of PBC_i plus the inactive nodes inside the I/O list*.

To determine the amount of main memory consumed by the approach of this section, we notice that this amount is equal to the space (S_1) occupied by the active nodes, plus the space occupied by the deactivated nodes inside the I/O list. Let S_2 be the amount of primary memory occupied by nodes deactivated during one time-period. As Lemma 3 states, the space occupied by nodes deactivated during one time-period, is O(*N*).

Lemma 3: The amount of primary memory occupied by the nodes deactivated during one time-period is O(*N*).

Proof: We attach a counter on every leaf of the active_PBs. When a new leaf is created, its counter is set to 0. Each incoming message may produce an update to at most two leaves, the leaf that contained the object, and the leaf that contains the object currently. If an update occurs inside a leaf, the counter of the leaf is increased by 1. Assume now that the *N* messages sent within the same time period have been processed (the I/O list is empty), and let *X* be the number of times that any counter has been increased by 1. Since every message increases by 1 at most two counters, it is obvious that $X \leq 2N$. From these *X* times that a counter has been increased, only $X / \Theta(B)$ times have led to a rebalancing operation, because of Property 2. Thus, the total number of rebalancing operations among the leaves of the active_PB's is $O(N/B)$. The nodes involved in these rebalancing operations are those that became inactive during the time period and, since each of them occupies O(*B*) space, we conclude that the total space occupied by them in primary memory is O(*N*). □

All we now need is to make sure that the hardware is capable of performing all the I/Os created during a time period, before the next time period ends (otherwise the I/O list would grow indefinitely, leading to a vast consumption of primary memory). Thus *R*, the maximum number of I/Os the hardware can perform, must be big enough. This is logical, because even if the number of I/Os caused by our solution is asymptotically optimal, we still need hardware that can handle this amount of I/Os, otherwise the solution will not work. This is why parameter *R* and the I/O list have been included in the solution, i.e. in order to indicate the hardware requirements. Apart from that, they add nothing to the solution. From Lemmas 2 and 3, it follows that, as long as $R = O(N/(T*B))$ we conclude that $M = S1+S2 = \Theta(N)$.

From the analysis of partially persistent B-trees ([2], [13]), it can easily be deduced that the secondary memory used is $O(W/B)$ blocks.

4.4 Spatiotemporal Range Predicates

As mentioned before, static indexing is suitable for the efficient processing of Spatio-temporal Range Predicates (*S*, *T*). The output is either the set of objects inside the spatial constraint *S* at the time instance *T*, or the set of objects inside *S* at some time instance during the time interval *T*. The efficient

processing of such a predicate by using a grid and a persistent structure for each cell of the grid was discussed in [9]. Here, we use the same solution, except from the fact that now we have to additionally search in primary memory.

Consider the predicate (S_1, T_1), where S_1 is the cell C_1 and T_1 is the time instance t_1. Let $F(C_1, t_1)$ be the set of objects satisfying the predicate (C_1, t_1). We then have to look in both the active_PBC_1 and in PBC_1, in order to retrieve all the records that correspond to objects that were inside cell C_1 at time instance t_1. Let D_1 and D_2 be the set of objects corresponding to the records retrieved from active_PBC_1 and PBC_1, respectively. Then $F(C_1, t_1) = D_1 \cup D_2$.

Now assume that the temporal constraint T_1 is the time interval $[t_1, t_2]$. To retrieve all the records corresponding to the objects that were inside the cell for a time instance in $[t_1, t_2]$, we have to look again in both the active_PBC_1 and in PBC_1. First, we access all the leaves that were active at time t_1. Then we can follow the *history* from time t_1 up to time t_2. The ability to follow the history is justified as follows: When one or two leaves of the index structure become inactive, either one or two new leaves are created. When a leaf L becomes inactive we store into it a pointer to the newly born leaf. If two new leaves are created, we store two pointers into L. Following the pointers we retrieve all leaves that were valid at some time instance during the interval $[t_1, t_2]$. These leaves contain the output of the query.

In [9] it is proved that, we can evaluate the spatiotemporal predicate (C_1, T_1), by sparing at most $O(\log_B WC_1 + F(C_1, T_1)/B))$ I/Os, where WC_1 is the number of updates occurred inside cell C_1.

5. Dynamic Indexing

The approach of Section 4 may not be the best if we are interested in nearest and k-nearest neighbour predicates. Such a predicate is a pair (Q, T), where Q is a point on the map and T can be either a time instance or a time interval. The output is the nearest object (or the k-nearest objects) to Q, at the time instance T or during the time interval T. The reason for the potential inefficiency of static indexing in this case is the following: If the query point is on an empty cell, we have to start searching the neighbouring cells. That is, in case of a sparse traffic, we will end up consuming too much time (one I/O per cell) discovering empty cells. To face the disadvantages of the grid approach, we need a more dynamic partitioning of the 2-d space. Thus, we divide our 2-d space by using only vertical lines. The area between two consecutive vertical lines is called *slab*, thus each slab is determined by its left and right border in the x-coordinate. Contrary to grid cells, slabs can easily be indexed in a way that their x-range is not static, i.e., the vertical lines that define slabs can “move” during the monitoring period. The reason is that by moving a line we have to update only 2 slabs whereas in Section 4, if we move a vertical line, we will have to update many cells. The reason for moving a line is to maintain “equally balanced” slabs. The definition of “equally balanced” slabs

follows: *If d is a constant, it is said that the slabs are equally balanced if the most populated slab has at most d times the number of objects of the least populated slab.*

Maintaining equally balanced slabs, we know that the slab containing the query point will always contain an object that is "fairly close" to the query point, and can be used as a starting-point in order to find the nearest neighbor. After finding a starting point, we are able to bound the search area for the nearest neighbor, by searching for objects that are closer to the query point than the starting-point. Each time we find a new nearest neighbor, we bound further the search area.

We index the objects inside each slab, by their y-coordinate. Assuming that the slabs are thin enough, the y-coordinate suffices to track the objects inside each slab with a satisfactory precision. If follows, that we can track the current position of the objects by a two-level indexing structure (Figure 3). The upper level is an index for the slabs. Each leaf of the upper level corresponds to a slab and it is connected to an indexing structure of the lower level, which stores the objects inside the slab, by their y-coordinate. If we want to track the past positions of the objects also, we have to make this two level indexing structure partially persistent.

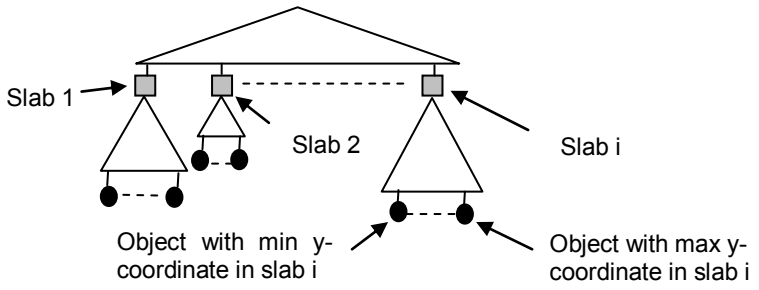


Fig. 3. The two-level partially persistent indexing structure.

To make things easier during the description of the approach we assume, as in Section 4, that when a message storing I/O is needed, it is performed immediately. In Subsection 5.4, we add an I/O list to the solution i.e., when a message storing I/O is needed, it is inserted into the I/O list that works in FIFO order. Finally, in Subsection 5.5, we explore the efficiency of the approach for the handling of nearest and k-nearest queries.

5.1 Data Structures

In secondary memory we maintain the two-level persistent indexing structure described above. We call this structure, *slab_index*. The upper part is a persistent B-tree, and the same also holds for each tree of the lower part.

In primary memory we maintain an indexing structure created by the active nodes of the slab_index. We call this structure *active_slab_index.* In primary memory we also maintain a table A, to store the objects. Position *i* of A stores a pointer to the record of object *i,* in the active_slab_index.

5.2 Splitting and Merging Slabs

A slab *has* to merge with another slab, if its objects reduce to N_L *-1* and it *has* to split if its objects increase to N_H*+1*. Parameters N_L and N_H are going to be determined later on. A slab that has more than N_H objects is called *big*, whereas a slab that has less than N_L objects is called *small*. A slab that is neither big nor small is called *normal*. The split and merge operations are incremental i.e., they are completed through small steps, where each such step costs a constant amount of time. Incremental split/merge steps are performed each time we receive a message from an object that is either inside or entered or left one of the involved slabs.

The incremental merge operation works as follows: Assume that slab S_i has N_L-1 objects. Let TR_i be the tree of the lower level of the *active_slab_index* that corresponds to S_i. Let S_j be the slab that is going to be merged with S_i, and TR_j be the lower level tree that corresponds to S_j. TR_i and TR_j have the same parent, u. Our objective is to merge trees TR_i and TR_j, incrementally. The merging procedure is straightforward. In particular, every incremental step merges *O(1)* objects from each tree, according to their y-coordinate, starting from the leftmost leaf of each tree. Each merged record is inserted into tree TR_k, which is going to replace trees TR_i and TR_j. Incremental merge steps are performed each time we receive a message from an object that is either inside or entered or left one of the two merging slabs.

Suppose now that the number of objects of slab S_k increased to N_H+1, i.e. we have to split S_k. Again, the split is incremental. Each incremental step, processes *O(1)* records of the lower level tree, TR_k, corresponding to S_k, and inserts them into a temporary balanced binary search tree, according to their x-coordinate. When all the records of T_k are inserted into the temporary tree, we start creating the two new slabs that are going to replace S_k. We incrementally traverse the leaves of the temporary tree (*O(1)* records per incremental step) from left to right. The left half data records enter the new lower level tree TR_{k1} that corresponds to the new slab S_{k1}. The right half data records enter the new lower level tree TR_{k2} that corresponds to the new slab S_{k2}. When the traversal is over, the leaf of the upper level corresponding to S_k

is split into two, and each new upper level leaf is connected with one new lower level tree.

During the incremental merge, an object O_i may have a valid record in at most two trees. The first record (the one pointed by the pointer in position A[*i*]) is inside a leaf (L_1) of a lower level tree (T_1) corresponding to an existing slab. The second record is inside a leaf (L_2) of a tree under creation (T_2). If a new message arrives from Object O_i, both leaves must be updated. We reach the record of O_i in L_2 by storing to the record of O_i in L_1, a pointer to L_2.

It remains to give the algorithm that triggers these split/merge operations. This algorithm, which is called *overall algorithm*, guarantees that there is an upper and a lower bound to the number of objects inside each slab. The problem that needs to be solved is the following: Assume that a slab S_i has N_L-1 objects, therefore it must merge with another slab. However, both neighboring (to S_i) slabs are under an incremental split/merge operation. Then, S_i must wait for at least one of these split/merge operations to finish. The overall algorithm must guarantee that S_i will not wait forever and furthermore, all slabs are "equally balanced". The main idea of the overall algorithm is the introduction of *critical* slabs. If S_i is found to be small and all the neighbouring to S_i slabs are under split/merge operations, then S_i becomes *critical*. From that point, and as long as S_i is critical, when an update occurs inside S_i, an incremental step is performed for each neighbouring (to S_i) split/merge operation. The overall algorithm follows.

Begin (overall algorithm)

We set $N_H = 10N_L$ and we also set for each incremental step, to process at least 65 active records. Suppose that we perform an update inside a slab S_i.

Step 1: If S_i is already under a split/merge operation, we perform an incremental step for this split/merge operation.

Step 2: If S_i is found to be big (i.e. if it has more than N_H objects), a split operation starts for S_i.

Step 3: If S_i is critical, then an incremental step is performed for each neighbouring (to S_i) split/merge operation.

Step 4: If S_i is found to be small (i.e. it has less than N_L objects), we have to find another slab to merge it with S_i. If there is a slab S_j next to S_i, such that S_j is not under a split/merge operation, then we merge S_i with S_j. Otherwise, S_i becomes "*critical*".

Step 5: If we have just executed the last incremental step for a merge operation and the resulting slab is big, (see Lemma 4) then the resulting slab immediately starts a split operation.

Step 6: If we have just executed the last incremental step for a merge operation and the resulting slab is normal, the resulting slab starts a merge operation with its neighbour that first became critical (if it has critical neighbours).

Step 7: If we have just executed the last incremental step for a split operation, then (according to Lemma 5), the resulting slabs are normal. Each of the resulting slabs merges with its critical neighbour, if such a neighbour exists.

End (overall algorithm)

Since every incremental step is executed each time an update occurs, it follows that the objects inside a slab may increase by 1 in every incremental step, if the update that caused the incremental step was an insertion. Thus, if L is the number of objects inside a slab when the slab begins to split, then the number of incremental steps needed for the split is at most L/64 (since we process at least 65 objects per incremental step and one object can be added per incremental step). Similarly, if the total number of objects inside a pair of slabs that start to merge is L then the merge operation will need at most L/64 incremental steps.

Lemma 4: A merge operation always creates either a big or a normal slab.

Proof: First of all, it is trivial to show that a merge operation may create a big slab. Assume that two slabs start to be merged. One of these slabs must be small, and the other must be non-big. We conclude that when the merge operation starts, the maximum number of involved objects is $11N_L$ -1. Even if, during the merge operation, the two slabs experience only deletions, then the resulting slab will have more than $11N_L - 1 - (11N_L - 1)/64 > 10N_L$ objects, i.e., it will be big.

We are now going to prove that a merge operation does not create a small slab. In order to do that, we are going to determine the minimum number of objects involved in a merge operation, according to the overall algorithm. Assume thus that a slab S_i becomes small, but all the neighbouring slabs are under a merge operation. Thus, S_i becomes critical, and at the time one of these merge operation ends, S_i has at least N_L-$11\mathrm{N_L}/64$ elements objects (each merge operation involves at most $11N_L$ objects). Let S_j be the slab that results from this merge operation. S_j is then merged with its neighbouring slab that first became critical, and this neighbouring slab may not be S_i. Thus, S_i remains critical until this new merge operation ends, and let S_k be the bucket that results from this merge operation. According to the overall algorithm, S_k will be merged with S_i, because if S_k has two critical neighbors, S_i is the one that first became critical. When S_k starts to be merged with S_i, S_i has at least N_L-$22N_L/64$ objects. Thus the minimum number of objects inside a slab, when the slab starts a merge operation is $\mathrm{N_L}$-$22N_L/64$. We conclude that the minimum number of objects involved in a merge operation, when the merge operation starts is $2(N_L$-$22N_L/64)$ objects. Even if during the merge operation, all the updates that occur inside the two slabs are deletions, it follows that the resulting slab has at least $2(N_L$-$22N_L/64)$ - $2(N_L$-$22N_L/64)/64 > 2N_L$-$3N_L/8 > N_L$ objects. Therefore, the resulting slab is not small. □

Lemma 5. A split operation always creates normal slabs.

Proof: First of all we have to determine the maximum number of elements inside a slab, when this slab starts to be split. If slab S_i drops to N_L-1 objects, it starts to merge with a slab S_j, which is not under a split process. When the incremental merging process completes, the resulting slab has at most $11N_L$+$11N_L/64$ objects (if S_j had N_H objects and only insertions occurred during the merge operation). Thus, the maximum number of elements inside a slab, when this slab starts to be split is $11N_L$+$11N_L/64$.

Now, we are going to show that a split operation creates non-big slabs. If the split starts with $11N_L$+$11N_L/64$ objects (and only insertions occur during

the split operation), then at the time the split ends, the number of objects may increase to $11N_L+11N_L/64 +(11N_L+11N_L/64)/64< 12N_L$ which means that each new slab has at most $6N_L$ objects (i.e., it is not big)

Let us now show that a split operation creates non-small slabs. Each split starts with more than $10N_L+1$ objects. If only deletions occur during the split operation, the resulting slabs will have more than $(10N_L-10N_L/64)/2 >4N_L$ elements. □

Lemma 6. The slabs are "equally balanced".

Proof: From Lemmas 4 and 5 it follows that the minimum number of objects inside a slab is $N_L-22N_L/64$, whereas the maximum number is less than $12N_L$. This means that the maximum number is less than is 19 times greater than the minimum number and according to Definition 3, Lemma 6 follows □

We have not set the value of N_L. From a theoretical point of view, any value is fine. From a practical point of view however, a very small value would create too thin slabs, leading to many split/merge operations between slabs, which will increase the number of I/Os (although they will still be asymptotically optimal). On the other hand, a big value would create too thick slabs, and this fact will have a negative effect on the efficiency of the system for answering queries. Thus, the value of N_L should be determined according to the above guidelines, through experiments.

A technical detail still remains in the dark. When a split (merge) operation completes, the lower level tree (trees) corresponding to old slab (slabs) becomes inactive. For each object inside the leaves of such (i.e., inactive) trees, we update the corresponding pointer in table A, to point to the correct position, which is a leaf of the new tree. The space occupied by the nodes of this tree cannot be released immediately. Some leaves of the tree may not be identical to their image leaves in secondary memory. These image leaves must be updated. This task is called *cleaning procedure*, and it is charged on the slabs created by the split/merge procedure. It is easy to see that the I/Os caused by the cleaning procedure do not asymptotically change the number of message storing I/Os, since a slab being cleaned, has experienced $O(N_L)$ updates, and the number of message storing I/Os needed to clean it is $O(N_L/B)$.

5.3 Algorithm to Handle Incoming Messages.

Suppose we receive a message (O_k, x_value, y-value, t).

Step 1. We search the active_slab_index in order to find the slab (which is a leaf of the upper level) containing that x_value. Let S_i be that slab. We search table A and find the record of the object. Moving upwards in the lower level tree containing that record, we reach its root, and therefore we find the previous slab of the record. Let S_j be the previous slab.

Step 2. If $i=j$, we reach the current record of O_k and update its y-coordinate. We do that by deactivating the current record of the object (we set its end-value to the current time instance), and inserting a new record for that object, with the current y-coordinate of that object as key. Then we

proceed to the update algorithm of the persistent structure. As in Section 4, if an active leaf becomes inactive we update its image leaf in the slab_index, and if a new active leaf is created, we create its image-leaf in the slab_index. If the record of an object is copied (as a result of the rebalancing of a leaf) to a new leaf, we update the pointer in A for that object. In particular, we store in A[k] a pointer pointing to the new record of object O_k.

Step 3. If i≠j, we delete (deactivate) the current record of the object in S_i and we insert a new record for that object in S_j. The insertion or deletion is performed as in step 2.

Step 4. We update S_i and S_j according to the overall algorithm.

5.4 Analysis of the Solution

Assuming that no slab is ever split or merged with another one, it can be easily derived using the arguments of Section 4 that the total number of message storing I/Os is O(W/B), i.e. property 1 holds. However, the existence of split/merge operations between slabs complicates things, because it is not now clear that Property 1 holds. All we need to show is that the total number of message storing I/Os because of the split/merge operations between slabs is also O(W/B). This is proved by Lemma 7.

Lemma 7: The total number of message storing I/Os created by the incremental split/merge operations between slabs is O(W/B).

Proof. We attach a counter on every slab. When a new slab is created, its counter is 0. If an update occurs inside a slab, the counter of the slab is increased by one. Therefore, each incoming message may produce an update into at most two slabs. After the W messages have been applied, let X be the number of times that any counter increased by 1. It is obvious that $X \leq 2W$. From these X times that a counter was increased, only $X/\Theta(N_L)$ split/merge operations occurred, because each split/merge operation "costs" $O(N_L)$ incoming messages (since each incoming message triggers an incremental split/merge step and each such step processes a constant number of data records). Thus, the received messages generate at most $O(W/N_L)$ split/merge operations. Each split/merge operation creates $\Theta(N_L/B)$ message storing I/Os. We conclude that the total number of message storing I/Os because of the incremental split/merge operations is $O(W/N_L)*\Theta(N_L/B)=O(W/B)$. □

The space occupied by our structures in secondary memory is O(W/B) blocks, as in Section 4. This can be easily derived by the analysis of partially persistent B-trees (see [2], [13]). Concerning the space occupied by the active nodes in primary memory, Lemma 2 continues to be satisfied and as a result, this amount of space is O(N). Assuming that each I/O is performed immediately, no additional amount of primary memory is needed. Otherwise, we can assume that the I/Os are performed by the use of an I/O list that works in a FIFO manner (as we have assumed in Section 4). In order to guarantee that the total amount of primary memory used is still O(N), all we need is to

make sure that the space occupied by inactive nodes corresponding to pending I/Os (inside the I/O list), is also O(*N*). Lemma 3 continues to hold and, as a result, we conclude that the amount of primary memory used remains O(*N*), as long as *R* = O(*N*/(*T***B*)).

5.5 Nearest and k-nearest Neighbor Queries

As mentioned in Section 2, by indexing the space in a dynamic way we are able to efficiently process Spatio-temporal *nearest (k-nearest) neighbor* predicates. Suppose we have to process the nearest neighbor predicate ((*x*, *y*), *T*), where *x*, *y* are the x- and y-coordinates of the query point and *T* is a temporal constraint (time instance or time interval).

Assume first that *T* is a time instance *t*. We find the slab containing (*x*, *y*) at time *t*, by searching the slab_index and the active_slab_index, using *x* and *t* as keys. When we reach the leaf of the upper level that is connected to the slab of interest, we proceed to the lower level tree. We search the lower level tree using *y* and *t* as keys. We extract O(*B*) records valid at time *t* and, from these records, we choose the closest to the query point. Let O_k be that object. Figure 4 shows a possible scenario. Observe that O_k, which is found to be the closest object to the query-point inside the slab that contains the query-point, is not actually the closest object. Indeed, it is O_i which is is closer. This "error", occurred because we use only the y-coordinate to index all the objects inside the same slab and, with respect to the y-coordinate, O_k is closer than O_i to the query point. Object O_k may not be the closest to the query point, but it can be used as a "pruning tool". In particular, we now know that there is no reason to search for the nearest object outside the circle in Figure 4. This circle enables us to search inside the gray area, because this is the best we can do according to our indexing structures. (We can afford to efficiently search between two y-coordinates inside each slab.)

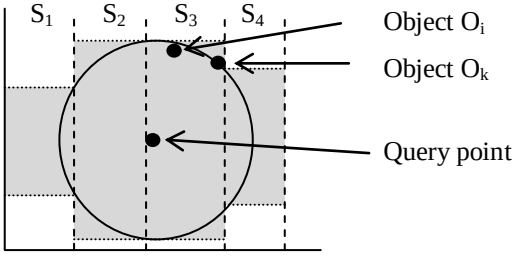


Fig. 4. O_k is not the closest object to the query point inside S_3, although it was retrieved as such by searching inside S_3

Observe that O_i, in Figure 4, may not in fact be the object closest to the query point, because slab S_2 may contain an object that is even closer. A search in S_2 will reveal the one closest**.** The advantage of the above described strategy is that we always find an initial object, which can be used

as a pruning tool (O_k in Figure 4), because all slabs have objects. This advantage may prove valuable in terms of time efficiency, especially in cases where the number of objects is small or in cases where the movement of the objects tend to form areas of very sparse traffic.

If the time constraint T is a time interval $[t_1, t_2]$, things are more complicated. Here is a general description. During time, we keep the "history" of the leaves by using pointers as explained in Subsection 4.4. We find the nearest neighbour to the query point, at time t_1, as described in the previous paragraph. Suppose that the nearest neighbour at time t_1 is object O_j (see Figure 5). Then, the area of interest contains the slabs intersected by the circle *(i.e.,* S_2 and S_3, between y_1 and y_2). For each slab, we must search in the interval $(t_1, t_2]$, for a potential new nearest neighbour. All we need to do is to "follow the time". This can easily be achieved by the use of pointers that lead, with respect to time, to the descendants of each leaf. Since at most two leaves can be created by a leaf rebalance, it follows that we can move forward in time, by storing two pointers in each leaf. We start from each leaf that contains elements in the gray area at time t_1, and follow the pointers leading to the future, until we reach leaves created after t_2. In each leaf we access, we search for a potential new nearest neighbor. Every time we find a new nearest neighbor, we update the slabs and areas of interest.

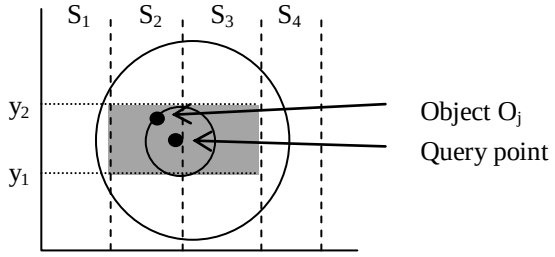


Fig. 5. The dark-shaded area is the area of interest created by O_j, which is the nearest neighbour of the query point at time t_1. Accessing the leaves that correspond to the gray area at time t_1, and moving forward in time until we reach t_2, we manage to retrieve the final (and truly) nearest neighbour.

We can use this solution to answer k-nearest neighbor queries as well. The only difference is that initially, we retrieve the k-nearest neighbors from the slab containing the query point. Then, the intersecting slabs and the ranges of interest in each slab, are determined via the k-th nearest neighbor after each I/O.

6. Dealing with the Rebalancing I/Os

In Sections 4 and 5 we assumed that no rebalancing I/Os occur, and this assumption is not realistic. Indeed a message storing I/O may trigger a non-constant number of rebalancing I/Os. However, in case that the tracking

period is longer than P (which is a realistic assumption), the amortized number of rebalancing I/Os per message storing I/O is constant. This is because the amortized update cost of the partially persistent B-tree is constant. As a consequence, Property 1 holds for the solutions of Sections 4 and 5, even if we take the rebalancing I/Os into account.

For a strictly theoretical solution (without the above realistic assumption), it suffices to guarantee that after a message storing I/O, O(1) rebalancing I/Os *in the worst case* may occur. Thus, to solve the problem, we need a partially persistent B-tree with constant worst case update time. Such a structure was recently presented (see [7]), and by adapting it in Sections 4 and 5 as our basic tree structure, we achieve O(1) rebalancing I/Os per message storing I/O, in the worst case.

7. Conclusions and future work

In this paper we studied the problem of storing the past and present positions of moving objects and proposed theoretical solutions for storing the messages sent by the moving objects in a way that guarantees that the number of the involved I/Os is asymptotically optimal. This way, we managed to face the bottleneck caused by the large number of messages that must be taken into account in order to update the involved indexing structures.

In particular, we presented solutions that require $O(W/B)$ I/Os for storing the messages (where W is the total number of messages and B is the number of messages that fit into a disk block) and require $O(N)$ primary memory (where N is the number of tracked objects). Furthermore, our solutions allow for the efficient processing of interesting queries. To show this, we focused on the processing of well-known spatiotemporal range predicates and nearest neighbor predicates, which have been studied thoroughly.

Exploring the efficiency of our solutions, in order to answer other interesting queries, is a promising field for future work. New queries emerge daily from real life demands and, for each such query, a solution based on the techniques we presented in this paper may turn out to be useful.

Also, the implementation of our techniques is another interesting piece of research. However, a strict implementation of the proposed solutions is not a very good idea. Such an implementation should definitely incorporate ideas from other implementation oriented research efforts in order for practical efficiency worthy of investigation, to be achieved.

Acknowledgement. The authors are most grateful to the reviewers, whose fruitful and constructive comments helped to substantially improve the readability of the paper.

References

1 Aggarwal, A., Vitter, J. S.: The input/output complexity of sorting and related problems. Communications of the ACM, Volume 31, Issue 9, 1116-1127. (1988)

2 Becker, B., Gschwind, S., Ohler, T., Seeger, B., Widmayer, P.: An asymptotically optimal multiversion B-tree. The VLDB Journal, Volume 5, Issue 4, 264-275. (1996)

3 Biveinis, L., Saltenis, S., Jensen C. S.: Main memory operation buffering for efficient R-tree update. In the Proceedings of the 33rd international conference on Very large data bases (VLDB), ACM, Vienna, Austria, 591-602. (2007)

4 Cheng, R., Xia, Y., Prabhakar, S., Shah, R.: Change Tolerant Indexing for Constantly Evolving Data. In the Proceedings of the 21st International Conference on Data Engineering (ICDE), IEEE Computer Society press, Tokyo, Japan, 391-402. (2005)

5 Driscoll, J. R., Sarnak, N., Sleator, D. D., and Tarjan, R. E.: Making Data Structures Persistent. Journal of Computer and System Sciences, Volume 38 Issue 1, 86-124. (1989)

6 Kwon, D., Lee, S., Lee, S.: Indexing the current positions of moving objects using the lazy update R-tree. In the Proceedings of the 3rd International Conference on Mobile Data Management (MDM), IEEE Computer Society, Singapore, 113-120. (2002)

7 Lagogiannis, G., Lorentzos, N.: Partially Persistent B-trees with Constant Worst Case Update Time. Computers and Electrical Engineering, Volume 38, Issue 2, 231-242. (2012)

8 Lagogiannis G., Lorentzos N. A., Sideridis A.B.: Indexing Moving Objects: A Real Time Approach. In the proceedings of the 3rd World Summit on the Knowledge Society (WSKS), Corfu, Greece, 421-426. (2010)

9 Lagogiannis, G., Lorentzos, N., Sioutas, S., Theodoridis, E.: A Time Efficient Indexing Scheme for Complex Spatiotemporal Retrieval. SIGMOD Record Volume 38, Issue 3, 11-16. (2009)

10 Lanka, S., Mays, E.: Fully Persistent B+-trees. In the Proceedings of the ACM SIGMOD Conference on Management of Data, Denver, USA, 426-435. (1991)

11 Lee, M. L., Hsu, W., Jensen, C. S., Cui, B., Teo, K. L.: Supporting Frequent Updates in R-Trees: A Bottom-Up Approach. In the Proceedings of the 29th international conference on Very large data bases (VLDB), Berlin, Germany, 608-619. (2003)

12 Tung, H., Ryu K.: One update for all moving objects at a timestamp. In the Proceedings of the Sixth IEEE International Conference on Computer and Information Technology (CIT), IEEE Computer Society, Seoul, 6. (2006)

13 Varman, P. J., Verma, R. M.: An Efficient Multiversion Access Structure. IEEE Transactions on Knowledge and Data Engineering, Volume 9, Issue 3, 391-409. (1997)

14 Xiong, X., Aref, G. W.: R-trees with Update Memos. In the Proceedings of the 22nd International Conference on Data Engineering (ICDE), IEEE Computer Society, Atlanta, USA, 22. (2006)

George Lagogiannis is a Lecturer at the Agricultural University of Athens. He received his Diploma (1995) his M.Sc. (2000) and his Ph.D. (2003) in Computer Science, from the Department of Computer Engineering and Informatics of the Technical School of the University of Patras. His research interests are data structures and algorithms.

Nikos Lorentzos is a Professor at the Agricultural University of Athens. He has a first Degree in Mathematics (National and Kapodistrian University of Athens, 1975), a Master's Degree in Computer Science (Queens College, CUNY, USA, 1981) and a Ph.D. Degree in Computer Science (Birkbeck College, University of London, 1988). His major research contribution is in Temporal, Spatial and Spatiotemporal Databases, and in the development of DSSs and Expert Systems in the forestry and agricultural domain.

Alexander B. Sideridis received a primary degree in Mathematics from the University of Athens, and an M.Sc. and Ph.D. from Brunel University. He is a Professor and Head of the Informatics Laboratory of the Agricultural University of Athens. His current research interests concern the design and implementation of Decision Support and Knowledge Based Systems

Received: September 27, 2011; Accepted: July 04, 2012.

DOI:10.2298/CSIS110617004S

Multi-sensor Data Fusion Based on Consistency Test and Sliding Window Variance Weighted Algorithm in Sensor Networks

Jian Shu[1,2], Ming Hong[1,3], Wei Zheng[1,3], Li-Min Sun[2,4], and Xu Ge[1,3]

[1] Internet of Things Technology Institute, Nanchang Hang Kong University, Nanchang China
shujian@jxjt.gov.cn
[2] School of Software, Nanchang Hang Kong University, Nanchang China
zhengwei_nchu@126.com
[3] School of Information Engineering, Nanchang Hang Kong University, Nanchang China
hong19860320@hotmail.com
[4] Institute of Software Chinese Academy of Science, Nanchang China
sunlimin@is.iscas.ac.cn

Abstract. In order to solve the problem that the accuracy of sensor data is reducing due to zero offset and the stability is decreasing in wireless sensor networks, a novel algorithm is proposed based on consistency test and sliding-windowed variance weighted. The internal noise is considered to be the main factor of the problem in this paper. And we can use consistency test method to diagnose whether the mean of sensor data is offset. So the abnormal data is amended or removed. Then, the result of fused data can be calculated by using sliding window variance weighted algorithm according to normal and amended data. Simulation results show that the misdiagnosis rate of the abnormal data can be reduced to 3% by using improved consistency test with the threshold set to [0.05, 0.15], so the abnormal sensor data can be diagnosed more accurately and the stability can be increased. The accuracy of the fused data can be improved effectively when the window length is set to 2. Under the condition that the abnormal sensor data has been amended or removed, the proposed algorithm has better performances on precision compared with other existing algorithms.

Keywords: wireless sensor networks, data fusion, consistency test, sliding window, variance weighted.

1. Introduction

Wireless sensor network (WSN) which is constituted by a large number of micro-sensor nodes deployed in the monitored area can sense, collect, and process the information of monitored objects in the coverage area. Then the processed data is sent to the observer through the multi-hop self-organized network [1]. Since the nodes are generally battery-powered and deployed in a harsh environment area, some of which are not available for human, it's unrealistic to replace battery for continuous power supply. The nodes will die as long as the energy is drained out. The network may work abnormal or even failure once some nodes are dead. Moreover, the external noises and internal noises can affect the accuracy of the sensor data. The external noises include electromagnetic radiation, temperature and pressures, and internal noises include the decrease of stability and the zero offset in some sensors which have been used for a long time. With the help of multi-sensor data fusion algorithms, the precision of data can be improved.

In order to solve the problem that the precision of data fusion is low due to zero drift and the drop of the stability for part of the sensor when multiple sensor nodes measuring on the same target. This paper introduces a multi-sensor data fusion method based on consistency test and sliding-windowed variance weighted in sensor networks. Firstly, we propose a sensor measurement model, and the model can simplify the core problem to the result of internal noise. Then we present a consistency test with the new confidence distance to diagnose whether the mean of internal noise is shift under the sliding window mode, so that the abnormal sensor which will lead to zero drift can be amended or removed conditionally. Finally, we make data fusion processing of the normal sensors measured value and some certain amended abnormal sensors by sliding window sample variance weighted method, so that more precise data can be obtained.

2. Related Studies

Data fusion in WSN is different from traditional ones since the ability of node is limited. Nodes are battery-powered, the ability of CPU is weak, and wireless communication is unstable. Therefore, Traditional complex and high energy-consuming fusion algorithms are not suitable for WSN. There exist some algorithms, such as weighted average algorithm, Kalman Filter [2] and Bayes estimation [3] to solve the problem.

Weighted average algorithm is widely used as data fusion in WSN since it is simple and easy. Literature [4] proposed a variance weighted algorithm, and proved that variance weighted estimator is minimum unbiased estimation value of mean variance. The algorithm seems simple, but the variances of all nodes need to be given firstly. Batch estimation algorithm is proposed in literature [5], it's a kind of weighted average algorithm. It divides all sensors to two batches, and then uses variance weighted algorithm for fusion after

This paper is sponsored by the National Nature Science Foundation of China (No.60773055), and Jiangxi Key Technology R&D Program (No.2009BGA01000).

calculating the sample mean and the sample variance of each batch. Adaptive variance-weighted method is proposed in literature [6] under the premise that external noise is stable. it is proposed to solve the problem that the variance of each sensor is unknown with the help of the variance of sample. Iteration method is used to aggregate multi-sensor data in literature [7], it is based on adaptive variance weighted algorithm. And the result shows that good precision can be obtained. Literature [8] adopts window variance weighted algorithm in this direction and illustrate its idea on window size definition according to different noise change. In regard to the characteristics of sensor noise abruption, Literature [9] proposed the variance weighted algorithm based on adaptive window length. It divided noise estimate curve into smooth zone and abrupt zone by detecting noise change in sensor data, meanwhile it uses corresponding window size to revise multi-sensor fusion value and improves final aggregating accuracy according to different curve level.

Consistency test method focuses on the problem that there will be a deviation when various types of sensors measuring on the same target, it tests and removes the sensor with larger deviation to reduce the impact on fusion. Nowadays there are mainly some consistency test methods based on relation matrix and distribution graph, and according to how to determine the relation matrix, the former method which is based on relation matrix can be divided into three parts:1. It is a consistency test method based on relation matrix which is determined by the confidence matrix[10], which is established on known measurement model and noise as the Gaussian noise, the relation matrix is obtained by calculating the confidence distance between each two nodes, and then, this method tests the sampling value with larger deviation by graph theory approach;2.It is a consistency test method based on relation matrix which is determined by degree of support[11].Based on the measurement model, this method quantifies the support degree of the measured value of each two sensors by an exponential decay function. And determines the sampling value with larger deviation by the experience threshold value; 3. It is a consistency test method based on relation matrix which is determined by statistic distance [12], the method is still established on known measurement model and noise as the Gaussian noise, defines the statistic distance of observations of each two sensors based on the multivariate normal distribution to determine the relation matrix, and obtains sensor set with the biggest mutually support through directed graph theory; 4. It is a consistency test method based on relation matrix which is determined by empirical threshold [13], the method determines the trust degree of observations of each two sensors by the curve function with a empirical threshold to obtain the relation matrix, and determines the weight of each sensor observations by the largest eigenvector of the matrix, finally makes the fusion processing. Relying on Moffat distance to define relational matrix, the consistency check approach [14] uses Moffat and involving criterions to compute both relational matrix and correlated graph and searches sensor group with a Max support degree by liner fit method.

3. System Model

In wireless sensor network, as show in Fig.1, there are discrepancies of the values on the same target measured by different sensors because of the affection from noises. The noises include external noise and internal noise.

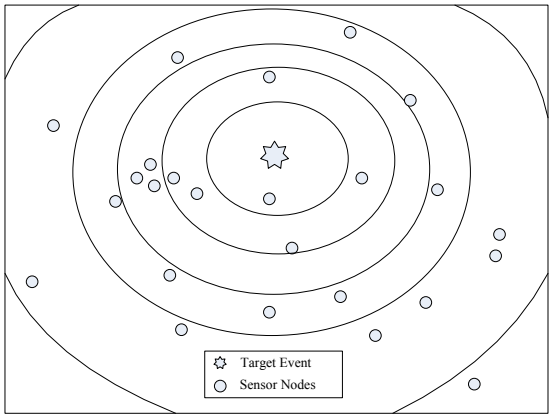


Fig. 1. A example of Sensor Nodes Deployment.

External noise is mainly caused by environment change which includes temperature, pressure and electromagnetic radiation. And we use Gaussian white noise which is zero mean value and different variance in the model definition period.

Internal noise is mainly caused by the sensors themselves. It is relatively stabilized, and it is not changed in a short period. For example, there exists zero offset because of shedding of element wiring, burn-in and similar factors. It is usually using Gaussian white noise which is nonzero mean value and constant variance in the model definition period. The zero drift phenomena are assumed as the measured values of some sensors are smaller or larger than normal ones. The decreasing stability of sensors is showed as the large undulatory property of the measured values.

Literature [15] proposes a sensor measuring model $z = x + \eta$ and a noise model $\eta \sim N(0,1)$. Considering the change of both external noise and the inconsistency of noise among different sensors, a new sensor measuring model is shown as formula (1).

$$z_i(k) = x(k) + \gamma(k) + \xi_i \qquad (1)$$

$z_i(k)$ is the $k-th$ measured value of sensor *i*. $x(k)$ is the $k-th$ real value of target object, and it is a constant if k is given . $\gamma(k) \sim N(0, \sigma^2(k))$ is the $k-th$ external noise, it is changed with times. $\xi_i \sim N(\mu_i, \sigma_i^2)$ is the internal noise of sensor *i*, it is stable. It is not changed with times in a short period.

Assuming that external noise and internal noise are mutual independent, the measuring model can be simplified as formula (2).

$$z_i(k) = x(k) + \eta_i(k) \tag{2}$$

$\eta_i(k) \sim N(\mu_i, \sigma^2(k) + \sigma_i^{\ 2})$ is integrated noise.

It is supposed that *n* sensors measure a same target simultaneously. Each sensor contains a sliding window whose length is W for storing sampling values in the first W times. The $k-th$ measured value of sensor *i* is $z_i(k)$. The sliding window's sample mean is $\bar{z}_i(k)$ and its sample variance is $S_i^{*2}(k)$.

4. Multi-sensor Data Fusion Based on Consistency Test and Sliding Window Variance Weighted Algorithm in Sensor Networks

4.1. The traditional consistency test algorithm

Luo and his assistants proposed a consistency test to solve the problem of inconsistence of measured value from the sensors which measure on the same target. Based on the sensor measuring model which is established in this algorithm, the error sensor data is removed after calculating the confidence distance of each two sensors and establishing the relationship matrix between the sensors, and then the optimal statistical decision making methods are used for fusion.

Confidence Distance [10]: There are *n* sensors which measure the same target, and the measured data of all sensors x_1, x_2, ..., x_n can be obtained, if x_1 follows Gaussian distribution and the corresponding density function is $P_i(x)$. The confidence distance can be obtained by formula (3).

$$d_{ij} = 2\left|\int_{x_j}^{x_i} P_i(x \mid x_i)dx\right| = 2A \tag{3}$$

$$P_i(x \mid x_i) = \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{1}{2}\left(\frac{x - x_i}{\sigma_i}\right)^2\right\} \tag{4}$$

Where d_{ij} in the formula (3) represents for the confidence distance between sensor *i* and sensor *j*, σ_i is the variance of sensor *i*, *A* is the area enclosed by the conditional probability density curve, $x = x_i$, $x = x_j$ and x-axis, shown in Fig. 2.

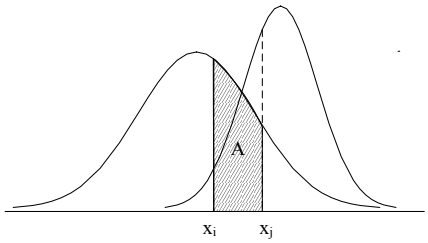


Fig. 2. The schematic diagram of confidence distance.

The smaller the value of d_{ij} is, the closer value of sensor *i* to sensor *j* is.

In order to simplify the calculation, Luo has introduced the error function (5), and d_{ij} is shown as formula (6).

$$erf(\theta) = \frac{2}{\sqrt{\pi}} \int_0^{\theta} \exp(-z^2) dz \tag{5}$$

$$d_{ij} = erf(\frac{x_j - x_i}{\sqrt{2}\delta_i}) \tag{6}$$

Confidence distance matrix [10]: The confidence distances of each two sensors can be obtained. They constitute the $n \times n$ matrix defined as the confidence distance matrix $D_{n \times n}$.

$$D_{n \times n} = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ d_{n1} & d_{n2} & \cdots & d_{nn} \end{bmatrix} \tag{7}$$

Relation Matrix [10]: Since the threshold d_{ij} is given, the relationship value of each two sensors can be calculated by formula (8). We constitute the $n \times n$ matrix defined as the relation matrix $R_{n \times n}$.

$$r_{ij} = \begin{cases} 1 & d_{ij} \le \varepsilon_{ij} \\ 0 & d_{ij} > \varepsilon_{ij} \end{cases} (i, j = 1,2,...,n) \tag{8}$$

$$R_{n \times n} = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{bmatrix} \tag{9}$$

In the relation matrix $R_{n \times n}$, r_{ij} represents for the support degree of the sensor *i* to sensor *j*. $r_{ij} = r_{ji} = 0$ expresses that sensor *i* and sensor *j* don't

have any relationship. When one of r_{ij} and r_{ji} equals 0 and another equals 1, it means that the relationship between them is weak. If $r_{ij} = r_{ji} = 1$, it means that sensor *i* and sensor *j* have a strong relationship. Thus, the largest supported sensor set is obtained through the relation matrix, and then the optimal estimation methods are used for the last aggregation.

But there are several problems to be solved in wireless sensor networks:

1) According to the confidence distance from formula (5), each sensor's variance is need to be given for the calculation of confidence distance of each two sensors, it is can be described by integrated noise variance $\sigma^2(k) + \sigma_i^2$, but it can be obtained in wireless sensor networks. So it is not suitable.
2) Since the confidence distance from formula (5) and (6) contains integral calculation, the calculation is so complex that it is not suitable in wireless sensor network.
3) The method doesn't propose a approach how to get the largest supported sensor set.

Therefore, we make the following improvements on the algorithm.

4.2. A new definition of confidence distance

According to formula (2), we can get the following conclusions: the difference between the values can be obtained by the $k-th$ data of sensor *i* and *j*, and it follow the Gaussian distribution, that is

$$z_i(k) - x(k) \sim N(\mu_i(k), \sigma^2(k) + \sigma_i^2) \qquad (10)$$
$$z_j(k) - x(k) \sim N(\mu_j(k), \sigma^2(k) + \sigma_j^2)$$

The problem which is to judge whether one of sensor *i* or *j* incur zero offset can be transformed into the problem of the hypothesis testing of the mean difference for two samples of normal distribution:

$$H_0 : \mu_i(k) - \mu_j(k) = 0 \qquad (11)$$
$$H_1 : \mu_i(k) - \mu_j(k) \neq 0$$

Under the significance level α, the test statistic T is obtained.

$$T = \frac{\left|[\overline{z_i(k) - x(k)} - \overline{z_j(k) - x(k)}] - 0\right|}{\sqrt{\frac{\sigma^2(k) + \sigma_i^2}{n_i} + \frac{\sigma^2(k) + \sigma_j^2}{n_j}}} \geq u_{1-\alpha/2} \qquad (12)$$

$x(k)$ is a constant, so formula(13) can be obtained.

$$T = \frac{\left|[\bar{z}_i(k) - \bar{z}_j(k)] - 0\right|}{\sqrt{\frac{\sigma^2(k) + \sigma_i^2}{n_i} + \frac{\sigma^2(k) + \sigma_j^2}{n_j}}} \geq u_{1-\alpha/2} \tag{13}$$

If the test statistic T meets the conditions, we can reject the hypothesis H_0, it represents that there is a large difference between sensor *i* and *j*, and one of them may exist zero offset.

But the external noise variance $\sigma^2(k)$ and internal noise variance $\sigma_i^2(k)$ of each sensor can not be obtained, and $\sigma_1^2(k)$, $\sigma_2^2(k)$, …, $\sigma_i^2(k)$ are not the same in wireless sensor networks.

In order to obtain the test statistic, we introduce the conclusions of the limit distribution, a new test statistic as shown as formula (14).

$$T = \frac{\left|\bar{z}_i(k) - \bar{z}_j(k)\right|}{\sqrt{\frac{S_i^{*2}(k) + S_j^{*2}(k)}{n}}} \geq u_{1-\alpha/2} \tag{14}$$

Therefore, we can define a new confidence distance as follow:

$$d_{ij} = d_{ji} = \alpha = 2[1 - P\{x \leq \frac{\left|\bar{z}_i(k) - \bar{z}_j(k)\right|}{\sqrt{\frac{S_i^{*2}(k) + S_j^{*2}(k)}{n}}}\}] \tag{15}$$

Where d_{ij} is the significance level of the hypothesis testing, according to the consequences of committing two type errors, when the value of α is small, the probability of error type II increases accordingly, that is it will be easy to make substandard products in the test sample judged to be qualified, then to accept the original hypothesis. If the value of α is large, the probability of error type I increases, so it is easy to make qualified products in the test sample is deemed to have failed and then to be refused. Considering that the abnormal sensor have much great impact on fusion, we need to minimize the probability of error type II, that means we should try our best to prevent abnormal sensors judged to be normal ones. Therefore, the larger the value of d_{ij} is, the less obvious the mean integrated noise of sensor *i* and *j* is. That is, sensor *i* and *j* may both belong to the normal sensors and may also both belong to abnormal sensors.

But formula (15) can not be applied in sensor nodes because of complex calculation. Therefore it requires an easy method to calculate sensor relational matrix directly. In order to solve the problem, the threshold ε_0 of significance level need be given firstly. The result can be obtained by the condition (16),

$$T_{ij} = T_{ji} = \frac{\left|\bar{z}_i(k) - \bar{z}_j(k)\right|}{\sqrt{\frac{S_i^{*2}(k) + S_j^{*2}(k)}{n_{window}}}} \geq u_{1-\varepsilon_0/2} \tag{16}$$

In corresponding to relational matrix factor $r_{ij} = r_{ji} = 0$,otherwise, $r_{ij} = r_{ji} = 1$. In this way, it can avoid complex calculation and reduce energy consumption.

4.3. The diagnosis of abnormal sensors

According to the new definition of confidence distance, the confidence matrix $D'_{n\times n}$ is obtained, and the relation matrix $R'_{n\times n}$ is also obtained.

Relation matrix $R'_{n\times n}$ is a symmetric matrix composed by 0 and 1. $r_{ij} = r_{ji} = 0$ represents that the mean integrated noise of sensor *i* is much different from sensor *j*, therefore, one of the sensors must be a abnormal sensor. $r_{ij} = r_{ji} = 1$ represents that the mean integrated noise of sensor *i* is less different from sensor *j*, that's they may be both normal or abnormal.

Assuming the sensor node is the vertexes of graph G and relational matrix $R'_{n\times n}$ is the adjacency matrix of graph G, hereby, we could plot the entire correlation graph of all sensor nodes. According to theory of resolving maximum clique G' of graph G[16], the vertexes of clique G' composed normal sensor group A, and the remaining of them composed abnormal sensor group B. In order to avoid judging mistakenly, it is necessary to make sure the percentage of normal sensors is beyond 50%. Otherwise, it is possible to make a wrong judgment.

Algorithm 1: program for Max Clique

```
MaxClique(G; size)
 if |G|=0 then
    if size>max then
        max:=size
        New record; save it.
     end if
     return
end if
while G !=0 do
     if size + |G|6max then
```

```
            return
        end if
        i:=min{j | vj ∈G}
        G:=G\{vi}
       MaxClique(G ∩ N(vi); size +1)
    end while
    return
```

Fig. 3 is the relationship diagram G showing the degree of support of 1-7 sensors, in which node 1, node 2, node 3 and node 4 constitute the maximum clique G'. Therefore, we can determine that node 1, node 2, node 3 and node 4 constitute the normal sensor set, and node 5, node 6, and node 7 constitute the abnormal sensor set.

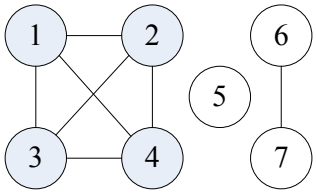


Fig. 3. The diagram of the degree of support for each sensor.

4.4. The sliding window variance weighted algorithm and how to amend or remove the measured data from abnormal sensors

The fundamental principle of adaptive weighted algorithm: under the condition of minimum average variance, it can find the best corresponding weight W_i of each sensor node with an adaptive way, and help $\hat{S}$ achieve the best fusion result. As.0 Fig.4 shows, S_i is the measure value of sensor nodes, where *i*=1,2…,*n*, while $\hat{S}$ is the final fusion result.

According to this theory, the sliding window variance weighted algorithm: In wireless sensor networks, since the external noise variance $\sigma^2(k)$ and the internal noise variance $\sigma_i^{\,2}(k)$ of the $k-th$ measurement of sensors carried by each sensor node are unknown. In order to solve this problem, the sample variance can be used to replace real variance, the weight of each sensor data can be obtained by formula (17).

$$W_i = \frac{1/S_i^{*2}(k)}{\sum_{j=1}^{n} 1/S_j^{*2}(k)} \qquad (17)$$

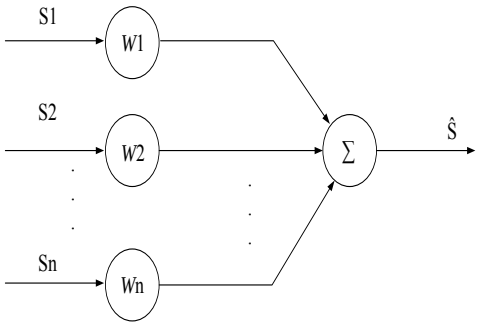


Fig. 4. Model of adaptive weighted estimate fusion.

```
Algorithm 2: program for sample variance
while(n<WINDOW_NUM) do
    if(Position<Size) then
       AVE     ←average value
       if(position=0) then
       VAR=0;  ←variance=0
       else caculate VAR
       New record; save it.
       end if
    end if
    else if(Position=Size) then
            caculate AVE
            caculate VAR
         end if
end while
return
Algorithm 3: program for sliding-window weight and fusion value
while(n<NODE_NUM) do
    if(G.pNode!=0)
       update SensorValue
       update SensorWeight
    end if
```

```
    else if(G.pNode!=0&&Node[i].VAR<Node[j].VAR)
      update SensorValue
      update SensorWeight
      end if
  end while
```

How to amend or remove the measured data from abnormal sensors: if the greatest normal sensor set *A* and abnormal sensor set *B* are obtained, we amend or remove under certain conditions:

1) Sensor $m \in B$, if $\exists$ sensor $n \in A$, and $S_m^{*2}(k) < S_n^{*2}(k)$, $z_m(k)$ is needed to be amended by using formula (18), (19) and (20).

$$z_m(k)' = z_m(k) - \Delta_m \tag{18}$$

$$\Delta_m = \bar{z}_m(k) - \sum_{i \in A} w_i \bar{z}_i(k) \tag{19}$$

$$w_i = \frac{1/S_i^{*2}(k)}{\sum_{j \in A} 1/S_j^{*2}(k)} \tag{20}$$

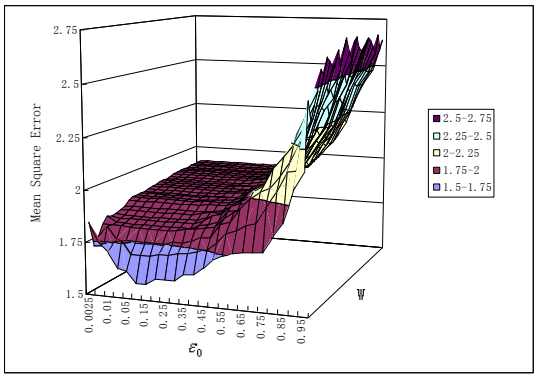
2) Sensor $m \in B$, if $\forall$ sensor $n \in A$, there is $S_m^{*2}(k) \geq S_n^{*2}(k)$, $z_m(k)$ is needed to be removed simply.

5. Simulation Research

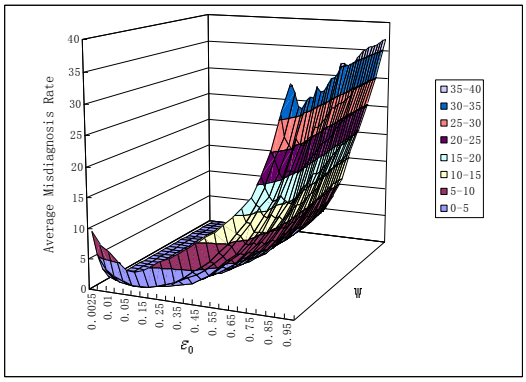
In order to verify the validity of the algorithm, OMNet++ is used for simulation. According to the experimental results obtained under different significance level threshold ε_0 and the length of window W, the optimal value ε_{best} and W_{best} can be evaluated. Then the precision of algorithm is compared with three other fusion algorithms such as arithmetic average algorithm, batch estimation algorithm and the adaptive variance weighted algorithm.
Cluster-based routing protocol is used in the experiment, each cluster has 41 nodes (including one cluster head), and the cluster head takes responsible for data fusion. The sliding window length of the member nodes is set to W, and significance level threshold is ε_0.

According to the model that $z_i(k) = x(k) + \gamma(k) + \xi_i$.Gaussian white noise whose mean is zero is used for simulating the external noise $\gamma(k)$, and its variance will change every R_γ times. Internal noise ξ_i can be described by Gaussian white noise whose mean is non-zero. The percent of abnormal

sensor node is P, it assumes that the internal noise is stable, and will not change as time changes. Considering the changing characteristic of the target object's actual value, $x(k)$ is generated randomly and changes every R_x.



(a) Mean Square Error.



(b) Average Misdiagnosis Rate.

Fig. 5. Simulation Results under Different Significance Level Threshold and Window Size W.

5.1. The best significance level threshold ε_{best}

In order to obtain optimize significance level threshold ε_{best}, the parameters are set as follows: n=40, N=200, R_γ=10, P=0.3, R_x=10, and $\varepsilon_0 \in [0,1]$ and $W \in [2,30]$. Fig. 5(a) and Fig. 5(b) illustrate the simulation results for this experiment. The x-axis represents significance level threshold and the y-axis represents window size. The z-axis represents mean square error in

the Fig. 5(a). The z-axis represents average misdiagnosis rate in the Fig. 5(b).

1) $\varepsilon_0 \in [0,0.05]$, mean square error and average misdiagnosis rate decreases rapidly when ε_0 is increasing. The higher the value of ε_0, the lower the probability of Error-type-I is, it means that the normal sensor nodes have less probability to be diagnosed as abnormal nodes.

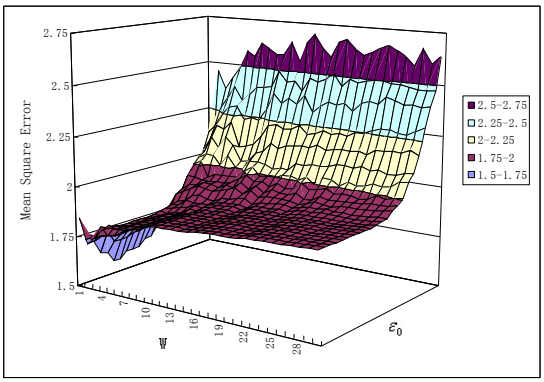
2) $\varepsilon_0 \in [0.05,0.15]$, mean square error and average misdiagnosis rate tend to be stationary. The reason is that when ε_0 changed within the range, all the abnormal sensors are diagnosed correctly, it has less effect on mean square error and average misdiagnosis rate.

3) $\varepsilon_0 \in [0.15,1]$, mean square error and average misdiagnosis rate increases rapidly as ε_0 is increasing. The higher the value of ε_0, the higher the probability of Error-type-II is. It means that the abnormal nodes can be mistakenly diagnosed as normal nodes easily.

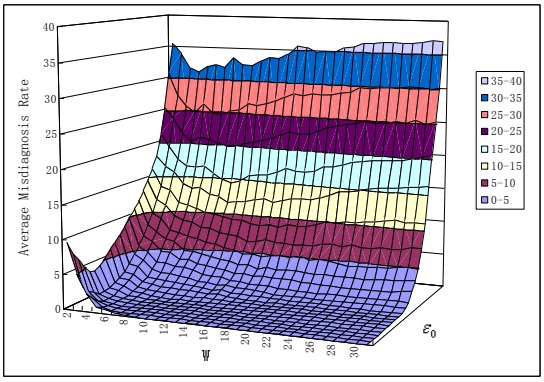
Therefore, the optimal range of significance level threshold ε_0 is [0.05, 0.15].

5.2. The best window size W_{best}

Similarly, in order to obtain the optimistic window size W_{best}, Fig. 6(a) and Fig. 6(b) illustrate the simulation results for this experiment. The x-axis represents window size and the y-axis represents significance level threshold. The z-axis represents mean square error in the Fig. 6(a). The z-axis represents average misdiagnosis rate in the Fig. 6(b).



(a) Mean Square Error.



(b) Average Misdiagnosis Rate.

Fig. 6. Simulation Results under Different Significance Level Threshold and Window Size W.

According to Fig. 6, the mean square error increases with the increment of window size, but the average misdiagnosis rate increases little. Thus, it is clear that window size W only have impact on the mean square error.

1) $W \in [2,10]$, the mean square error rises sharply. Because the window size is in the range of R_x and R_γ. The larger window size is, The lower fusion accuracy is.

2) $W \in [10,30]$, the mean square error rise gradually, because the window size already exceeds both R_x and R_γ. The deviation between estimated sensor noise variance and real one reaches the highest value.

Therefore, the optimistic sliding window size W_{best}=2 is obtained.

5.3. The precision comparison

Fig. 7 shows the simulation results of the proposed algorithm compared with arithmetic average algorithm, batch estimation algorithm and adaptive variance weighted algorithm under the condition of n=30, W=W_{best}=2, N=100, ε_0 = ε_{best} =0.10, R_γ =10, P =0.3 and R_x =10.

According to Fig. 7, traditional adaptive variance weighted approach has the Max average variance result. Because it relies only on sample variance as the candidate criterion of estimating whether the sensor nodes are normal or not, and adopts weighted mean theory and uses sample variance as the weight to obtain data fusion result, while ignores the influence on data fusion procedure triggered by Zero offset. For example, if the target sensors emerge Zero offset but the stability of its measurement value is on good condition,

this method will make the weight too big from its possible value range, directly cause fusion precision depressing and lead to the worst fusion result.

Batched estimation algorithm only uses batching method to fuse multi-sensor data, takes the reciprocal of each sample variance as the weight and sample average value and neglects the fact that sensor nodes lacks stability. Experiment result shows that average variance tends to high which means a low data fusion outcome; and arithmetic mean algorithm improve the fusion precision compared to above two approaches by using average calculating operation that may cover those effects brought by Zero offset and low stability sensors, for the weights of sensor nodes are all equal. However, the fusion result remains undesirable for it don't take the problems of Zero offset and stability difference into account.

Based on traditional adaptive variance weighted algorithm, our protocol uses conditional amend method to correct some of the abnormal sensor value which have a good stability, then adds them into the weighted fusion sequence of normal sensor group making the information required by data fusion large enough to enhance fusion precision. And the result appears to have minimum average variance, estimated fusion data closer to real value and best fusion performance compared to others. The reasons are shown as follows: Firstly, consistency test is used for diagnosing the abnormal sensor data; Secondly, it corrects the abnormal data in some degree. Finally, sliding window weighted algorithm is used for the last fusion.

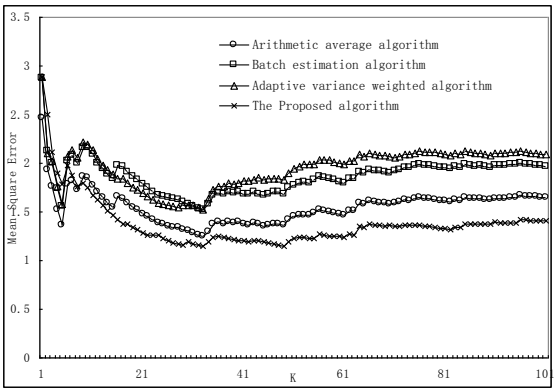


Fig. 7. The Precision Comparison using the Proposed Algorithm vs Arithmetic Average Algorithm, Batch Estimation Algorithm and Adaptive Variance Weighted Algorithm.

6. Conclusion

A novel sensor model and a new data fusion algorithm are proposed to solve the problem that the precision of measured value is low due to external noise

and internal noise. Firstly, an improved consistency test algorithm is used for diagnosing sensor data and obtaining the normal sensor set and abnormal sensor set. Secondly, the abnormal sensor value is amended or removed under some degrees. Finally, sliding window variance weighted algorithm is proposed for the last data fusion. The simulation result shows that the optimum consistency test threshold range is [0.05, 0.15] and the optimum sliding window size is 2. The results also show that it has better performances on precision compared with other existing algorithms.

References

1. Chong C Y, Kumar S P, “Sensor networks: Evolution, opportunities and challenges,” Proceedings of the IEEE, vol. 91, pp. 1247-1256, 2003.
2. Olfati-Saber R, “Distributed Kalman filter with embedded consensus filters,” Proceedings of Conference on Joint CDC-ECC’05, Dec 2005.
3. Zhang Shu-kui, Cui Zhi-ming, Gong Sheng-rong, “A Data Fusion Algorithm Based on Bayes Sequential Estimation for Wireless Sensor Network,” Journal of Electronics & Information Technology, vol. 31, pp. 716-721, 2009.
4. Liao Xi-chun, Qiu Min, Mai Han-rong, “Study on data fusion algorithms based on parameter-estimation,” Transducer and Microsystem Technologies, vol. 25, pp. 70-73, 2006
5. Zhang Xi-liang, Sun You, “A data aggregation arithmetic based on directed diffusion and batch estimate for wireless sensor network,” Control & Automation, vol. 6, pp. 173-174,180, 2006.
6. Zhong Chong-quan, Dong Xi-lu, Zhang Li-yong, “On the Estimation of Variances for Multi-Sensor Measurement,” Journal of Data Acquisition & Processing, vol. 18, pp. 412-417, 2003.
7. Liu Yuan-ze, Zhang Jia-wei, Li Ming-bao, “Support degree and adaptive weighted spatial-temporal fusion algorithm of multi-sensor,” CCDC ’10, pp. 4129-4133, 2010.
8. Xiao Long-yuan, Zeng Chao, “Adaptive second data fusion algorithm,” Transducer and Microsystem Technologies, vol. 26, pp. 81-83,180, 2007.
9. Zhang Yi, Jia Min-ping, “The Application of Variance-Estimation Based on the Adaptive Window Length in Multi-Sensor Data Fusion ,” Chinese Journal of Sensors and Actuators, vol. 21, pp. 1398-1401, 2008.
10. Luo R.C.,et al. “Dynamic multisensor data fusion system for intelligent robots”. IEEE Journal of Robotics and Automation,vol. 4, pp. 386-396, 1988.
11. Zhang Jie, Zhang Zong-Lin, Jing Bo, Sun Yong, “Spatial-Temporal Information Fusion for Multi-Source Node Cluster Based on D-S Evidential Theory and Fuzzy Integral of Support Degree ,” Chinese Journal of Sensors and Actuators, Vol.19 No.6 P.2727-2731, 2006.
12. Duan Zhan-Sheng, Han Chong-Zhao, Tao Tang-Fei, “Consistent Multi-sensor Data Fusion Based on Nearest Statistical Distance ,” Chinese Journal of Scientific Instrument, vol. 26 No.5, pp. 478-481, 2005.
13. Zhai Jian-She, Li Na, Wu Qing, “Improved Algorithm of Clustering-based Data Fusion for WSN ,” Chinese Journal of Computer Engineering, Vol. 34 No.11, pp. 134-136, 2008.

14. Mihaela Duta, Manus Henry, “The Fusion of Redundant SEVA Measurements,” IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, Vol.13, PP.173-184. 2005
15. Niu R, “Decision fusion in a wireless sensor network with a large number of sensors,” Proc. of the Seventh International Conference on Information Fusion, June 2004.
16. Pattic R., J Ostergard, “A fast algorithm for the maximum clique problem,” Discrete Applied Mathematics, vol. 12, pp. 197-207, 2002.

Jian Shu is a professor in school of Software, Nanchang Hangkong University, China. He received a Ms. in Computer Networks from Northwestern Polytechnical University in 1990. His research interests include wireless sensor network, embedded system and software engineering. He is the director of Internet of the Things Technology Institute, Nanchang Hangkong University.

Ming Hong is a postgraduate student of Nanchang Hangkong University. His research interest is Wireless Sensor Network.

Wei Zheng is a lecturer in school of Software, Nanchang Hangkong University, China. He received a Ph. D in Computer science and technology from Xidian University in 2010. His research interests include Wireless Sensor Network, optical network, network optimization and intelligence algorithm.

Li-Min Sun is a researcher in Chinese Academy of Sciences, Software Laboratories. He received his Ph. D from National University of Defense Technology, China. His research interest is wireless sensor network and mobile Ad Hoc network.

Xu Ge is a postgraduate student of Nanchang Hangkong University. His research interest is Wireless Sensor Network.

Received: June 17, 2011; Accepted: November 21, 2012.

DOI:10.2298/CSIS110613005Y

A Novel Method for Data Conflict Resolution using Multiple Rules

Zhang Yong-Xin[1], Li Qing-Zhong[2], and Peng Zhao-Hui[2]

[1]School of Mathematical Sciences, Shandong Normal University,
Jinan 250358, China
waterzyx@gmail.com
[2]School of Computer Science and Technology, Shandong University,
Jinan 250101, China
{lqz; pzh}@sdu.edu.cn

Abstract. In data integration, data conflict resolution is the crucial issue which is closely correlated with the quality of integrated data. Current research focuses on resolving data conflict on single attribute, which does not consider not only the conflict degree of different attributes but also the interrelationship of data conflict resolution on different attributes, and it can reduce the accuracy of resolution results. This paper proposes a novel two-stage data conflict resolution based on Markov Logic Networks. Our approach can divide attributes according to their conflict degree, then resolves data conflicts in the following two steps: (1)For the week conflicting attributes, we exploit a few common rules to resolve data conflicts, such rules as voting and mutual implication between facts. (2)Then, we resolve the strong conflicting attributes based on results from the first step. In this step, additional rules are added in rules set, such rules as inter-dependency between sources and facts, mutual dependency between sources and the influence of week conflicting attributes to strong conflicting attributes. Experimental results using a large number of real-world data collected from two domains show that the proposed approach can significantly improve the accuracy of data conflict resolution.

Keywords: Data integration, Data conflict resolution, Markov Logic Networks.

1. Introduction

Data integration is the process of providing users of an integrated information system with a unified view of several data sources. However, due to data quality discrepancy of data sources, different sources can often provide conflicting data; some can reflect real world while some cannot. To provide high-quality data to user, it is essential for data integration system to resolve data conflicts and discover the true values from false ones. This process is

called data conflict resolution and has recently received increasing attention in data integration field [1, 2, 3].

The current major works to resolve data conflicts are based on relational algebra and define some conflict resolution strategies and functions [4]. By relational operations expansion or user-defined-functions, user or domain expert can assign conflict resolution functions to different data conflicts according to their requirements or domain knowledge [5]. Though these methods can resolve data conflict to some extent, they fall short in the following aspects.

When new data and data sources are integrated into system, the previous assignment may be refined. Even a new conflict resolution function will be assigned or defined. So these methods can hardly adapt the situation where data integration is dynamic.

Among all conflict resolution strategies, “Trust your friends” and “Cry with wolves” [4] are widely used. Their principles are taking the value of a preferred source and taking the most frequent value. However, it is a challenge for data integration how to choose the most trustworthy data source. And it is arbitrary to only trust a certain source. In addition, especially on Web, with the ease of publishing and spreading information, the false information becomes universal. The voting strategy that prefers the most often frequent is not sufficiently reasonable. So the current methods can hardly guarantee the accuracy of data conflict resolution.

Current research focuses on resolving data conflict on single attribute, which does not consider not only the conflict degree of different attributes but also the interrelationship of data conflict resolution on different attributes, and it can reduce the accuracy of resolution results.

Recently, there has been a few interesting techniques developed that aim to identify the true values from false ones [6, 7, 8]. They can be called truth discovery or others. These approaches treat data conflict resolution as an inferring problem, and incorporate more semantic features and sophisticated human knowledge to determine which value is true. In the process of handling data conflicts, any helpful confidences and rules can be considered. However, as the uncertainty of the knowledge, it is a hot potato how to combine these evidences to infer the true values.

To adapt to dynamic data integration and incorporate uncertain knowledge to better resolve data conflict, a two-stage data conflict resolution based on Markov Logic Networks (MLNs) [9] is proposed. In Summary, we make the following three contributions:

We propose a two-stage data conflict resolution based on Markov Logic Networks. Our approach can divide attributes according to their conflict degree and separately handle conflicts on them in two stages. Because we consider the influence of week conflicting attributes to strong conflicting ones, this approach can improve the accuracy effectively.

Through observing and analyzing the characteristics of conflicting data and data sources, we extract and use multi-angle features and rules for true value inference.

Experimental results using a large number of real-world data collected from two domains show that the proposed approach can effectively combine these features and rules and significantly improve the accuracy of data conflict resolution.

This paper is organized as follows. We briefly review some related research efforts in Section 2, and describe the problem in Section 3. The overview of the proposed approach is introduced in Section 4, and the model details are described in Section 5. Experimental evaluations are reported in Section 6, and in the last section we draw conclusions and point out some future directions.

2. Related Work

The current major works to resolve data conflict on query time are based on relational algebra. The most representative work is conducted by Felix Naumann *et al*. Naumann *et al*. summarize current conflict resolution strategies and functions, and propose two research prototypes: HumMer [10] and FeSum [11]. They also extend and implement some relational operators such as *minimum union* [12].

Besides resolving data conflicts by relation expansion, there are some researches which focus on identifying true value from conflicting data. Minji Wu *et al*. [6] propose aggregating query results from general search engineer by considering importance and similarity of the sources. The importance of the sources can be measured by their ranks and popularity [13]. However, the rank of web pages according to authority based on hyperlinks does not reflect accuracy of information exactly. In addition, the method has certain limitation because it can only focus on queries whose answers are numerical values.

For discovering the true fact from conflict information provided by multiple data sources, Xiaoxin Yin *et al*. [7] propose an iterative algorithm - *TruthFinder*, which considers trustworthy of sources, accuracy of facts and interrelationship of two aspects. Nevertheless, this method does not consider dependence between sources in truth discovery. With the ease of publishing and spreading false information on the Web, a false value can be spread through copying and that makes truth discovery extremely tricky.

Xin Dong *et al*. [8] propose a novel approach that considers dependence between data sources in truth discovery. And they apply Bayesian analysis to decide dependence between sources [14] and design an algorithm that iteratively detects dependence and discovers truth from conflicting information. However, Bayesian model will be re-trained when some new inference rules join. So the approach is not adaptive enough.

In addition, the methods above mainly resolve data conflict on single attribute and do not consider not only the conflict degree of different attributes but also the interrelationship of data conflict resolution on different attributes. Thus, it can reduce the accuracy of resolution results.

Markov logic networks [9] is a simple approach to combining first-order logic and probabilistic graphical models in a single representation. As a general probabilistic model for modeling relational data, MLNs have been applied to joint inference under different domains, such as entity resolution [15] and information extraction [16, 17]. We will give a more detailed introduction to MLNs in Section 5.

3. Problem Definition

To make a clear presentation and facilitate the following discussions, we first explain some concepts in this paper in this section.

Data Source. The source which provides conflict information, such as databases, web sites, etc. A set of data sources can be represented as $S=\{s_1, s_2, \cdots, s_n\}$, where $s_i\ (1 \le i \le n)$ is the i^{th} data source.

Entity. An entity is a real world thing which is recognized as being capable of an independent existence and which can be uniquely identified, such as a book, a movie, etc.

Entity Attribute. Obviously, an entity attribute represents a particular aspect of a real world entity, such as an author of a book, a director of a movie. A set of entity attributes can be expressed as $A=\{a_1, a_2, \cdots, a_m\}$, where $a_i\ (1 \le i \le m)$ is the i^{th} entity attribute.

Fact. For an entity attribute, the value provided by a data source can be called fact. For example, for an entity attribute a (the author of book 'Flash CS3: The Missing Manual'), the data source s (the online book store 'ABC Books') provides a fact f ('Chris Grover, E. A. Vander Veer').

Data Conflict. When some data sources provide different facts for the same entity attribute, data conflict will be appeared.

True Value. In the conflicting facts, the fact which describes the real world is the true value.

Different data sources can provide different facts for some entity attributes. Among facts provided for an entity attribute, one correctly describes the real world and is the true value, and the rest are false. Fig. 1 depicts the sources, facts, entities, entity attributes and the relationships between them.

Definition 1. To input data source set S, entity set E, entity attribute set EA, fact set F and the relationships of them. For an entity attribute $ea \in EA$, $F=\{f_1, f_2, \cdots, f_{|F|}\}$ denotes the facts provided by S on ea and data conflict resolution is the process of identifying the true value f_i from F for each entity attribute, where $f_i \in F$.

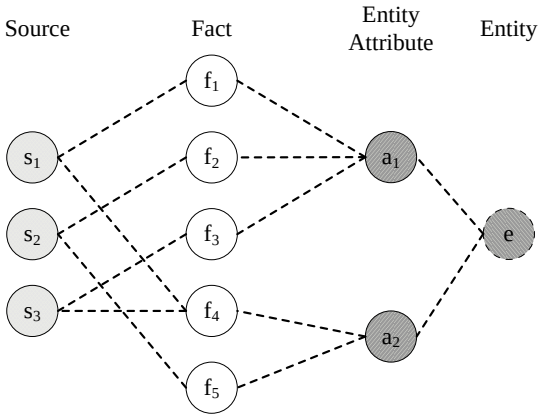


Fig. 1. Sources, facts, entities, entity attributes and the relationships

4. Approach Overview

In this paper, we propose a two-stage data conflict resolution based on Markov Logic Networks and the flowchart of our approach is illustrated in Fig. 2.

(1)First, data conflicting degree will be calculated on different attributes. According to conflicting degree, attributes can be divided into two sets: week conflicting attributes and strong conflicting ones.

(2)Then, data conflicts on week conflicting attributes will be resolved in the first stage. For resolving conflicts on these attribute, we use some rules such as voting and mutual implication between facts to train our MLN model with training set and infer the true values. Since the conflicting degree is low, resolution results will be highly accurate only through these simple rules.

(3)In the second stage, the results from the first stage can be added to the previous training set and our MLN model can be trained again with the new training set and inference can be carried out for the strong conflicting attributes. As the conflicting degree is high, more powerful rules will be added such as inter-dependency between sources and facts, mutual dependency between sources and the influence of week conflicting attributes to strong conflicting attributes. These rules can contribute to utilizing the resolution results from the first stage and improving the accuracy of data conflict resolution.

(4)Finally, we merge the data according to the inference results and can get accurate and consistent data set.

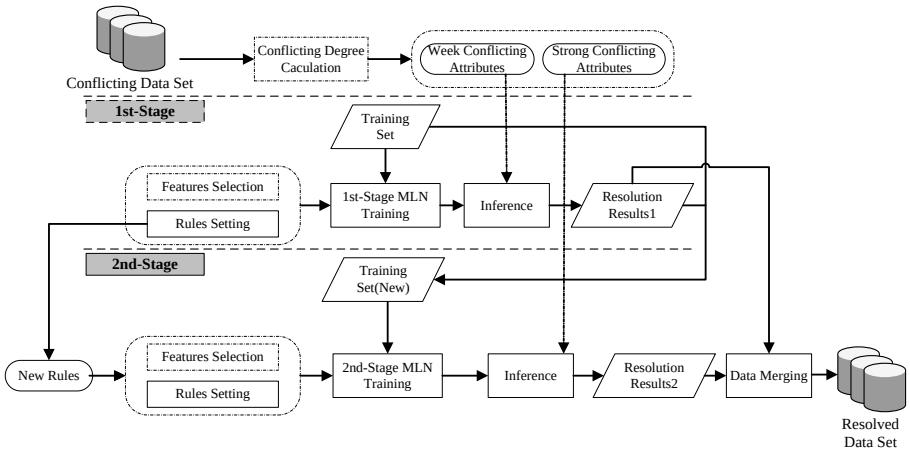


Fig. 2. The flowchart to the proposed approach.

The input of data conflict resolution is the data set with duplicates from different data sources, where the duplicates have been detected. And, the output is the data set in which the data conflicts have been resolved. The whole algorithm of data conflict resolution is showed below:

Algorithm 1.

Input: Integrated data set with duplicates D_C which contains attribute set A and entity set E. The training set is D_{Train} and the test set is D_{Test}. T is the threshold of data conflict resolution.

Output: The data set D_R whose data conflicts have been resolved.

$A_L := \varnothing, A_H := \varnothing$; // A_L, A_H denote separately week conflicting attribute set and strong one.
$D_R := \varnothing$; //resolved data set
for $a_i \in A$ do
  if $Conflict(a_i) < T$ then
    $A_L := A_L \bigcup \{a_i\}$
  else
    $A_H := A_H \bigcup \{a_i\}$;
  Define predictors and formulas, train our MLN model on D_{Train}. Infer the true values for A_L on D_{Test}, then get the result set D_1;
  $D_{Train} := D_{Train} \bigcup D_1$;
  Add new formulas and re-train our MLN model on D_{Train}. Infer the true values for A_H on D_{Test}, then get the result set D_2;

```
  for e_i ∈ E do
    for a_j ∈ A do
      Select true values according to D_1 and D_2, and
constitute a record r_i;
    D_R:=D_R ∪ {r_i};
  return D_R.
```

5. Model Detail

5.1. Conflicting Degree Measure

In integrated data set, different attributes have different conflicting degree. As described in Table 1, we collect information about the book “Flash CS3: The Missing Manual” (ISBN: 0596510446), which contains information about three attribute: Source, Title and Authors. We can see clearly that the titles of the book from different sources are accord with each other and the conflicting degree is low. However, the authors information is vary more widely and the conflicting degree is high. Obviously, if data sources provide more different facts, the entity attribute is more uncertain and the conflicting degree of it is higher. So we give the definition of conflicting degree of an entity attribute using information entropy.

Definition 2. For an entity attribute $ea \in EA$, let $F = \{f_1, f_2, \cdots, f_L\}$ be the fact set provided by different sources and $|f_i|$ denotes the frequency of the fact $f_i (1 \le i \le L)$, and then the conflicting degree of the entity attribute ea can be defined as follow:

$$EAConflict(ea) = -\sum_{i=1}^{L} p(f_i) \log p(f_i) \quad (1)$$

where $p(f_i)$ is the probability of the fact f_i, and $p(f_i) = \dfrac{|f_i|}{\sum_{j=1}^{L} |f_j|}$.

Definition 3. For an attribute $a \in A$, let $EA = \{ea_1, ea_2, \cdots ea_k\}$ be the corresponding entity attribute set, and then the conflicting degree of the attribute a can be defined as follow:

$$Conflict(a) = \frac{\sum_{i=1}^{k} EAConflict(ea_i)}{k} \quad (2)$$

Table 1. Conflicting information of a book

Source	Title	Authors
ABC Books	Flash CS3: The Missing Manual	Chris Grover, E. A. Vander Veer
A1 Books	Flash CS3: The Missing Manual	Veer, E. A. Vander, Grover, Chris
Auriga Ltd	Flash CS3: The Missing Manual	E A Vander Veer, Chris Grover, Vander Veer E., Grover Chris
textbooksNow	Flash CS3: Missing Manual	Vander Veer
Powell's Books	Flash Cs3: The Missing Manual	Vander Veer, E A
Book Lovers USA	Flash CS3: the Missing Manual, by Moore	Moore, Emily
Stratford Books	FLASH CS3	Glover

5.2. Markov Logic Networks

Markov Logic Networks (MLNs) [9] is a simple approach to combining first-order logic [18] and probabilistic graphical models in a single representation, and is a probabilistic extension of a first-order logic for modeling relation data. In MLNs, each formula has an associated weight to show how strong a constraint is: the higher the weight is, the greater the difference in log probability between a world that satisfies the formula and one that does not, vice versa. In this sense, MLNs soften the constraints of a first order logic. That is, when a world violates one formula it is less probable, but not impossible. Thus, for the problem of data conflict resolution, MLNs is a sounder model since the real world is full of uncertainty, noise imperfect and contradictory knowledge.

Definition 4. A Markov logic network L is a set of pairs $\{(F_i, w_i)\}_{i=1}^{m}$, where F_i is a formula in first logic and the real number w_i is the weight of the formula. Together with a MLN L and a finite set of constants $C = \{C_1, C_2, \cdots, C_{|C|}\}$, it constructs a Markov Random Field [19] $M_{L,C}$ as follows:

(1) $M_{L,C}$ contains one binary node for each possible grounding of each predicate appearing in L. The value of the node is 1 if the ground atom is true and 0 otherwise.

(2) $M_{L,C}$ contains one feature for each possible grounding of each formula F_i in L. The value of this feature is 1 if the ground formula is true and 0 otherwise. The weight of the feature is the w_i associated with F_i in L.

Thus, MLN can be viewed as a template for constructing Markov Random Fields [19]. The probability of a state x in a MLN can be given by:

$$P(X=x)=\frac{1}{Z}\exp\left(\sum_i w_i n_i(x)\right)=\frac{1}{Z}\prod_i \phi_i\left(x_{\{i\}}\right)^{n_i(x)} \quad (3)$$

where Z is a normalization factor employed for scaling values of $P(X=x)$ to [0,1] interval, $n_i(x)$ is the number of true groundings of F_i in x, $x_{\{i\}}$ is the state of the atoms appearing in F_i, and $\phi_i\left(x_{\{i\}}\right)=e^{w_i}$, w_i is the weight of the i^{th} formula.

Eq. 3 defines a generative MLN model, that is, it defines the joint probability of all the predicates. In our application of data conflict resolution, we know the evidence predicates and the query predicates a prior. Thus, we turn to the discriminative MLN. Discriminative models have the great advantage of incorporating arbitrary useful features and have shown great promise as compared to generative models [9,20]. We partition the predicates into two sets - the evidence predicates X and the query predicates Q. Given an instance x, the discriminative MLN defines a conditional distribution as follows:

$$P(q\mid x)=\frac{1}{Z_x(w)}\exp\left(\sum_{i\in F_Q}\sum_{j\in G_i} w_i g_j(q,x)\right) \quad (4)$$

where $Z_x(w)$ is the normalization factor, F_Q is the set of formulas with at least one grounding involving a query predicate, and G_i is the set of ground formulas of the i^{th} first-order formula. $g_j(q,x)$ is a binary function and equals to 1 if the j^{th} ground formula is true and 0 otherwise.

The problem of data conflict resolution introduced in this paper is to examine the correctness of conflicting facts and identify the true value corresponding to the real world. Thus, in our MLN model, we only need to define one query predictor as $IsAccurate(fact)$, which describe the accuracy of a fact. The confidence predictors can be the feature of conflicting facts. In a discriminative MLN model as defined in Eq. 4, the evidence x can be arbitrary useful features. With the predefined features, we define some rules or the formulas in MLNs. With these rules, MLN can learn the weight of the roles and infer the accuracy of facts.

5.3. Features

According to the observation and analysis of the features of sources and data, we extract features from the following four aspects: basic features, inter-dependency between sources and facts, mutual implication between facts and mutual dependency between sources. In the following, we will represent the above four kinds of evidences respectively and these features are presented as predictors in MLN model.

Basic Features

The basic features show source, entities, entity attributes, facts and the relationship between them. For example, a data source s provide a fact f, this evidence can be presented as $Provide(s, f)$. Also, to present the evidence that fact is a fact f about an entity attribute ea, we define a predictor $About(f, ea)$. In addition, for introducing the following voting rule, we introduce another evidence predictor $MaxFrequence(ea, f)$, which show that f is the most frequent fact about entity attribute ea.

Inter-dependency between Sources and Facts

Intuitively, there exists the "trustworthy" data source that frequently provides more accurate facts than other sources. This can be validated in the table I, which the data sources *ABC Books* and *A1 Books* are more trustworthy. And then, a fact is likely to be true if it is provided by trustworthy sources (especially if by many of them). Moreover, a data source is trustworthy if most facts it provides are true. Thus, we represent the trustworthy of a source and the accuracy of a fact as $IsTrustworthy(s)$, $IsAccurate(f)$ respectively.

Mutual Implication between Facts

Different facts about the same entity attribute may be conflicting. However, sometimes facts may be supportive to each other although they are slightly different. For example, for the book "Flash CS3: The Missing Manual", one data source claims the author to be "Chris Grover, E. A. Vander Veer" and another one claims "Veer, E. A. Vander, Grover, Chris". Though the expressions are different, two facts are equal. For another example, if two sources provide two facts: "E. A. Vander Veer" and "Vander Veer", then the content of the first fact contain the second one and the last one actually supports the last one. In order to represent such relationships, we represent them as $Equal(f_1, f_2)$ and $Contain(f_1, f_2)$.

Mutual Dependency between Sources

If two data sources provide many same facts for many entity attributes, then the two sources will be dependent each other, so the facts provided by them for others entity attributes may have the same accuracy. To describe the mutual dependency between sources, we define a predictor

$InterDepend(s_1, s_2)$. To describe the relationship more formally, we give the definition of the mutual dependency between sources.

Definition 5. For two data sources s_1, s_2, if they satisfy the equation $\frac{|F_1 \bigcap F_2|}{|EA_1 \bigcap EA_2|} \geq \alpha$, then there exists a dependency between the two data sources. Where F_1 and F_2 represent the set of facts provided by s_1, s_2 respectively, EA_1 and EA_2 represent the set of entity attributes for which s_1, s_2 provide the facts, and the threshold $\alpha \in [0,1]$. Sepcially, we regard two facts as equal only if they provide the equal value for the same entity attribute.

Table 2. The proposed features

Type	Feature	Description
Basic Features	$Provide(s, f)$	Data Source s provides the fact f.
	$About(f, ea)$	The fact f is about an entity attribute ea.
	$Belong(ea, e)$	ea is an entity attribute of entity e.
	$MaxFrenquency(ea, f)$	f is the most frequent fact among the facts about ea.
Inter-Dependency between sources and facts	$IsAccurate(f)$	The fact f is accurate.
	$IsTrustworthy(s)$	Data source s is trustworthy.
Mutual Implication between facts	$Equal(f_1, f_2)$	The two facts f_1 and f_2 have the same content.
	$Contain(f_1, f_2)$	The content of f_1 contains the one of f_2.
Mutual dependency between sources	$Depend(s_1, s_2)$	There exists mutual dependency between two data source s_1 and s_2.

5.4. Rules

Based on common sense and our observations on real data, we introduce the detail rules in this section. These rules show the heuristic characteristic and

are represented as predictor formulas in MLN. Because of the powerful and flexible knowledge representation, when new rules join, we can conveniently define new formulas to describe the rules and learn weights of the formulas to infer. Therefore, it makes our approach more adaptive. In addition, a majority of rules introduced in this paper are uncertain, and MLNs can handle uncertainty. Thus, any rules which are useful to resolve data conflict can be introduced to our approach even if the rules are imperfect and contradictory.

Rules in 1st Stage

In the first stage of data conflict resolution on week conflicting attributes, since the conflicting degree is low, we can get very high accuracy only through some simple rules. We will introduce voting rule and the rule of mutual implication between facts as follow.

Rule1: Voting

For the problem of identifying the true value from conflicting facts, voting is a naïve rule. Usually, the most frequent fact for an entity attribute is accurate.

$$MaxFrequence(ea, f) \Rightarrow IsAccurate(f) \tag{5}$$

Rule2: Mutual Implication between Facts

If two facts have the same content for an entity attribute ea, they have the same accuracy. As a rule the detailed information is better than the simple one. Thus, if the content of a fact f_1 contains the one of another fact f_2 and f_2 is accurate, then f_1 is also accurate.

$$Equal(f_1, f_2) \Rightarrow (IsAccurate(f_1) \Leftrightarrow IsAccurate(f_2)) \tag{6}$$

$$About(f_1, ea) \wedge About(f_2, ea) \wedge Contain(f_1, f_2) \wedge IsAccurate(f_2) \Rightarrow IsAccurate(f_1) \tag{7}$$

Rules in 2nd Stage

In the second stage of data conflict resolution on strong conflicting attributes, we add some more complex rules in our MLN model in order to utilize the resolved result from the first stage and handle the more strong conflicts. These rules include inter-dependency between sources and facts, mutual dependency between sources and influence of week conflicting attributes to strong ones.

Rule3: Inter-dependency between Sources and Facts

Base on analysis in the previous section, often the data source which provides accurate facts is trustworthy and the fact provided by trustworthy data sources is accurate. Therefore, we introduce the following formulas:

$$IsAccurate(f) \wedge Provide(s, f) \Rightarrow IsTrustworthy(s) \quad (8)$$

$$IsTrustworthy(s) \wedge Provide(s, f) \Rightarrow IsAccurate(f) \quad (9)$$

Rule4: Mutual Dependency between Sources

If two data sources provide many same facts for many entity attributes, there exists mutual dependency between the two sources. Therefore, the facts provided by them for other entity attributes likely have the same accuracy.

$$\begin{aligned} &InterDepend(s_1, s_2) \wedge About(f_1, ea) \wedge About(f_2, ea) \wedge \\ &\quad Provide(s_1, f_1) \wedge Provide(s_2, f_2) \\ &\Rightarrow (IsAccurate(f_1) \Leftrightarrow IsAccurate(f_2)) \end{aligned} \quad (10)$$

Rule5: Influence of Week Conflicting Attributes to Strong Ones

For an entity, if a data source provides true facts for many entity attributes, the facts provided by it for other entity attributes are probably accurate.

$$\begin{aligned} &Provide(s, f_1) \wedge About(f_1, ea_1) \wedge Belong(ea_1, e) \wedge \\ &Provide(s, f_2) \wedge About(f_2, ea_2) \wedge Belong(ea_2, e) \wedge \\ &\quad IsAccurate(f_1) \Rightarrow IsAccurate(f_2) \end{aligned} \quad (11)$$

5.5. MLN Weight Training and Inference

In addition to the features and formulas, a MLN must also include the relative weights of each of these clauses. However, in our case we do not know the relative strength of all of the above formulas beforehand. Therefore, we must train the model to automatically learn the weights of each formula.

The state-of-the-art discriminative weight learning algorithm for MLNs is the *voted perceptron* algorithm [21, 22]. The voted perceptron is a gradient descent algorithm that will first set all the weights to zero. It will iterate through the training data and update the weights of each of the formulas based on whether the predicted value of the training set matches the true value. Finally, to prevent over-fitting, we will use the average weights of each iteration rather than the final weights. In order to train the data using the voted perceptron algorithm, we must know the expected number of true groundings of each clause. This problem is generally intractable, and therefore, the MC-SAT [23] algorithm is used for approximation.

After learning the weights of the formulas, inference in MLN can be conducted. Traditionally, MCMC [24] algorithms have been used for

inference in probabilistic models, and satisfiability algorithms have been used for pure logical systems. Since a MLN contains both probabilistic and deterministic dependencies, neither will give good results. In our experiments, the MC-SAT algorithm will be used to determine the values of query predicates. The MC-SAT is an algorithm that combines MCMC and satisfiability techniques, and therefore performs well in MLN inferences.

Finally, according to the true value of each entity attribute, we merge all record referring to an entity to a single record. So we can get the result set.

6. Experiments Evaluation

We perform experiments on two real data sets to examine the accuracy of our approach. Our MLN model will be developed using the Alchemy system [25], which is an open source software package developed at the University of Washington that provides interfaces and algorithms for modeling Markov Logic Networks. In order to examine the effectiveness of our model, we perform experiments in the following aspects: (1) The accuracy of data conflict resolution; (2) The effects of changing the size of the training sample; (3) The effectiveness of two-stage data conflict resolution; (4) The effects of rules and their combination.

6.1. Datasets

Books

First, we extract book information from O'Reilly web site (http://oreilly.com/), including the book title, the authors, the publication date and ISBN. The data set contains 1,258 books and we regard it as ground truth (Our data set does not contain information from O'Reilly). Then, for each book, we use its ISBN to search on www.abebooks.com, which returns the online bookstores that sell the book and the book information from each store. We develop a program to crawl and extract the book information and get 26,891 listings from 881 bookstores. Since the ISBNs do not conflict each other, we perform our method to resolve the data conflicts about the book title, the authors and the publication date. In addition, we do a pre-cleaning of authors' names in order to remove some noise information.

Movies

In books data set, since the publication dates unlikely conflict each other for a same book, our method mainly resolve character data conflict, such as the

book title and the authors. To validate the ability of our method for resolving various type data conflict, we collect data about movies and examine the method for numerical data such as movie runtime. First, we extract top 250 movie information from IMDB.com, including the movie name, the directors and the movie runtime. Because of the authority of IMDB, we consider the information it provides as the standard facts (Also, information from IMDB.com is excluded from our data set). Then, according to the name of movies, we collect information of each movie using Google as described in [7]. The movies data set contains 7,119 movie listings from 952 data sources.

6.2. Experimental Results

Accuracy of Data Conflict Resolution

We measure the performance of data conflict resolution via accuracy, which can be defined as the percentage of the entity attributes whose true values are identified correctly over all entity attributes. We compare the accuracy of our approach against voting and *TruthFinder* [7] in the above two data sets. Our approach is represented as 2-Stage MLN. Specially, *TruthFinder* will give the incomplete facts partial scores. However, in our method, the incomplete facts can be considered as false. In addition, if two facts are equal, then the representation of them can be ignored. For example, for authors of a book, if the number of authors and each author's information are correct, then the authors' fact is correct, without considering the sequence of authors.

For the books data set, we randomly select records referring to 600 entities as training set. According to the conflicting degree, we select the book title and the publication date as week conflicting attributes, and execute first stage data conflict resolution utilizing our MLN model. In the second stage, we handle the strong conflicting attribute, i.e. the authors. In the movies data set, the training set contains records referring to 120 entities. By calculating the conflicting degree, we divide the directors and the movie runtime as the week conflicting attribute and the strong conflicting attribute respectively, and then execute 2-Stage data conflict resolution. In the experiments, we set the threshold for mutual dependency between sources as $\alpha = 0.8$, and the threshold for conflicting degree is set as $T = 0.5$.

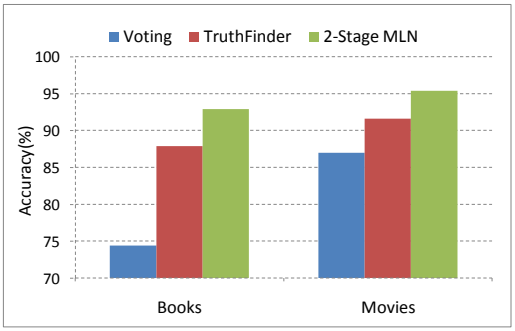


Fig. 3. Accuracy comparation among Voting, *TruthFinder* and our approach.

Fig.3 shows that our approach gets higher accuracy over other two approaches across the two data sets. In the books data set, our approach has an obvious advantage (the accuracy is 92.9%), it is because there exists plenty of incomplete or incorrect information for the book authors. It also validates the ability of our approach for resolving data conflict to some extent. But, our approach only gets a little higher accuracy than *TruthFinder* in the second data set. It is because that the movie runtime referring to a movie are not such variable as the book authors. And Voting also can get a high accuracy (87%). The experiments prove that our approach improve effectively the precision by 2-Stage data conflict resolution and utilizing multi-dimensional features.

Effects of Changing the Training Size

To check the effect factors of our approach, we test the effectiveness of the size of training samples. In the books data set, we randomly select records referring to 300, 600, 900, 1200 entities as training samples and resolve data conflict utilizing our approach. Otherwise, in the movies data set, records referring to 60, 120, 180, 240 entities are selected.

Fig.4 and Fig.5 show the accuracy with increasing training sizes on the books and movies data set respectively. When increasing the training size, a gradual improvement on accuracy is obtained. More interesting, the slope of the two curves becomes flatter and flatter as increasing the training size. It shows that the bigger of training size, the more precise of our approach. But with the training size is bigger and bigger, its effectiveness will degrade gradually. In addition, when the training size is too large, labeling the training sample will be time-consuming and it is not practical.

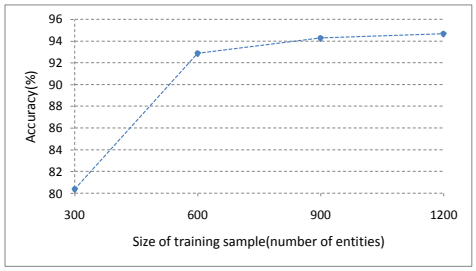


Fig. 4. Effects of changing the training size (The books data set)

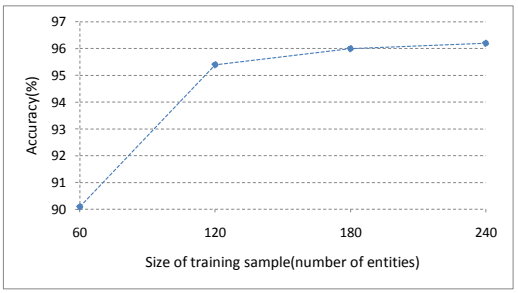


Fig. 5. Effects of changing training size (The movies data set)

Effectiveness of Two-Stage Data Conflict Resolution

One of the most important characteristics of our approach is to divide attributes into two sets according to conflicting degree and resolve data conflicts in two stages based on MLN. To validate the effectiveness of two-stage data conflict resolution, we conduct experiments as follows. First, we equally treat all attributes and resolve data conflicts in one stage, we call this approach as one stage data conflict resolution with MLN and denote it as 1-Stage MLN. As the rule of influence of week conflicting attributes to strong ones cannot be considered, we only use the first four rules in this approach. Then we resolve data conflict using our approach proposed in this paper and denoted it as 2-Stage MLN. We compare the accuracy of the two approaches.

Fig. 6 shows the comparation of accuracy of 1-Stage MLN and 2-Stage MLN in two data sets. 2-Stage MLN significantly improve accuracy of data conflict resolution compared to 1-Stage MLN. First, the first stage data conflict resolution can get highly accurate result for the week conflicting attributes. The result from the first stage effectively expands the training set in the second stage. And we utilize the more rules such as influence of week

conflicting attributes to strong ones and re-train the more precise MLN model to infer true values. Thus, the accuracy of resolution is improved effectively.

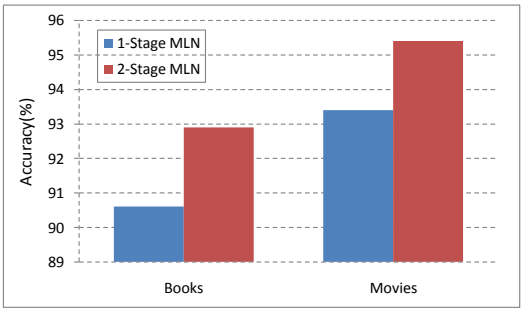


Fig. 6. Effectiveness of two-stage data conflict resolution.

Effects of Rules and Their Combination

To validate the rules proposed in this paper, the performance of our approach utilizing various rules and their combination is reported. The fifth rule needs to resolve data conflict on week conflicting attribute in advance and can be used only in the second stage, and the effectiveness of it has been validated in the third experiment. So we only test our four rules: Voting (denoted as *V*), Mutual implication between facts (denoted as *I*), Inter-dependency between sources and facts (denoted as *SF*), Mutual dependency between sources (denoted as *D*). We regard Voting as a basic rule, and then add one of the other three rules to the basic rule; finally we combine all the four rules. Thus, we get five rules and their combination. We test the accuracy of our approach utilizing the five respectively.

This experiment is executed in the books data set, and the other setting is the same as the first experiment. In Fig.7, it shows the accuracy using various rules and their combination. Obviously, each rule can improve the accuracy to some degree and it can validate the effectiveness of our rules. Among all rules, *I* and *SF* have a more obvious effect than *D*. On the one hand, it validates the existence of “trustworthy” data sources and the effect for identifying the true values from conflicting facts. On the other hand, conflicting information is often represented as incomplete or inconsistent, and it is one of the main troubles for resolving data conflict. In addition, we do not consider the dependency direction of *D*; it causes *D* not show enough significance.

This experiment also shows that our approach can combine various rules conveniently by adding or removing the corresponding formulas. Because data integration is a dynamic process, the appearance of new data conflict types can be predicted. We can extract new features and rules from new data

conflict types, and then MLN weight training and inference are conducted. It also demonstrates the adaptability of our approach.

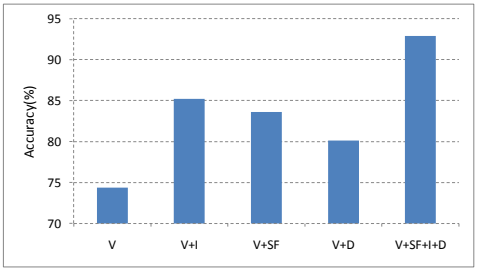


Fig. 7. Effects of rules and their combination.

7. Conclusion

In this paper we have presented an approach for two-stage resolving data conflict based on Markov Logic Networks. Our approach can divide attributes according to their conflict degree and separately handle conflicts on them in two stages. With multi-angle features and rules, our approach can effectively improve the accuracy of data conflict resolution. Based on the flexibility of knowledge representation as well as the ability to handle uncertainty of MLN, our approach can combine the imperfect and contradictory knowledge and is more adaptive. However, the training process of our model is something time-consuming when the training set is very large scale. So how to improve the efficiency of our approach is one of our future works.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China under Grant No. 90818001 and the Natural Science Foundation of Shandong Province of China under Grant No. 2009ZRB019YT and No. 2009ZRB019RW.

References

1. Dong, X., Naumann, F.: Data fusion – Resolving data conflicts for integration. In Proceedings of the 35th International Conference on Very Large Databases, Lyon, France, 1654-1655. (2009)
2. Galland, A., Abiteboul, S., Marian, A., Senellart, P.: Corroborating information from disagreeing views. In Proceedings of the Third International Conference on Web Search and Web Data Mining, New York, USA, 131-140. (2010)

3. Gatterbauer, W., Suciu, D.: Data conflict resolution using trust mappings. In Proceedings. of ACM SIGMOD International Conference on Management of Data, Indianapolis, Indiana, USA, 219-230. (2010)
4. Bleiholder, J., Naumann, F.: Conflict handling strategies in an integrated information system. In Proceedings of the International Workshop on Information Integration on the Web, Edinburgh, UK. (2006)
5. Bleiholder, J., Naumann, F.: Data fusion. ACM Computing Surveys. Vol. 41, No. 1, 1-41. (2008)
6. Wu, M-J., Marian, A.: Corroborating answers from multiple web sources. In Proceedings of the 10th International Workshop on the Web and Databases. Beijing, China. (2007)
7. Yin, X-X., Han, J-W., Yu, P. S.: Truth discovery with multiple conflicting information providers on the Web. In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Jose, California, USA, 1048-1052. (2007)
8. Dong, X., Berti-Equille, L., Srivastava, D.: Integrating conflicting data: the role of source dependence. In Proceedings of the 35th International Conference on Very Large Databases. Lyon, France, 550-561. (2009)
9. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning. Vol. 62, No. 1-2, 107-136. (2006)
10. Bilke, A., Bleiholder, J., Bohm, C., Draba, K., Naumann, F., Andweis, M.: Automatic data fusion with HumMer. In Proceedings of the 31st International Conference on Very Large Databases. Trondheim, Norway, 1251-1254. (2005)
11. Bleiholder, J., Draba, K., Naumann, F.: FuSem – Exploring different semantics of data fusion. In Proceedings of the 33rd International Conference on Very Large Databases. Vienna, Austria, 1350-1353. (2007)
12. Bleiholder, J., Szott, S., Herschel, M., Kaufer, F., Naumann, F.: Subsumption and complementation as data fusion operators. In Proceedings of 13th International Conference on Extending Database Technology. Lausanne, Switzerland, 513-524. (2010)
13. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: bringing order to the web. Technical report, Stanford Digital Library Technologies Project. (1998)
14. Berti-Equille, L., Sarma, A. D., Dong, X., Marian, A., Srivastava, D.: Sailing the information ocean with awareness of currents: Discovery and application of source dependence. In Proceedings of the 4th Biennial Conference on Innovative Data Systems Research. Asilomar, CA, USA. (2009)
15. Singla, P., Domingos, P.: Entity resolution with Markov logic. In Proceedings of the 6th Industrial Conference on Data Mining. Hong Kong, China, 572-582. (2006)
16. Yang, J-M., Cai, Y., Wang, Y., Zhu, J., Zhang, L., Ma, W-Y.: Incorporating site-level knowledge to extract structured data from web forums. In Proceedings of the 18th International Conference on World Wide Web. Madrid, Spain, 181-190. (2009)
17. Zhu, J., Nie, Z., Liu, X., Zhang, B., Wen, J-R.: Statsnowball: a statistical approach to extracting entity relationships. In Proceedings of the 18th International Conference on World Wide Web, Madrid, Spain, 101-110. (2009)
18. Genesereth, M. R., Nilsson, N. J.: Logical Foundations of Artificial Intelligence. Morgan Kaufmann, San Mateo, CA. (1987)
19. Poon H., Domingos, P.: Joint inference in information extraction. In Proceedings of 22nd AAAI Conference on Artificial Intelligence. Vancouver, Canada, 913-918. (2007)
20. Singla, P., Domingos, P.: Discriminative training of Markov logic networks. In Proceedings of the 20th National Conference on Artificial Intelligence. Pittsburgh, Pennsylvania, USA, 868-873. (2005)

21. Lowd, D., Domingos, P.: Efficient Weight Learning for Markov Logic Networks. In Proceedings of 11th European Conference on Principles and Practice of Knowledge Discovery in Databases. Warsaw, Poland, 200-211. (2007)
22. Collins, M.: Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In Proceedings of EMNLP, Philadelphia, PA. (2002)
23. Poon, H., Domingos, P.: Sound and efficient Inference with Probabilistic and Deterministic Dependencies. In Proceedings of the 21st National Conference on Artificial Intelligence. Boston, Massachusetts, USA, 458-463. (2006)
24. Gilks, W. R., Richardson, S., Spiegelhalter, D. J.: Markov Chain Monte Carlo in Practice. Chapman and Hall. London, UK. (1996)
25. Kok, S., Singla, P., Richardson, M., Domingos, P.: The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA. (2005). http://www.cs.-washington.edu/ai/alchemy.

ZHANG Yong-Xin, born in 1978, received the Ph. D. degree from School of Computer Science and Technology, Shandong University, Jinan, China, in 2012. Now, he is working at School of Mathematical Sciences, Shandong Normal University. His research interests include web data integration and web data fusion.

LI Qing-Zhong is a professor in School of Computer Science and Technology, Shandong University, China. His research interests include data integration and Software as a Service (SaaS).

PENG Zhao-Hui, born in 1978, Ph. D., associate professor. His research interests include keyword search in database and web data management.

Received: June 13, 2011; Accepted: November 08, 2012.

DOI:10.2298/CSIS111231001V

Ontology-Based Architecture with Recommendation Strategy in Java Tutoring System

Boban Vesin[1], Mirjana Ivanović[2], Aleksandra Klašnja-Milićević[1] and Zoran Budimac[2]

[1]Higher School of Professional Business Studies, Novi Sad
Vladimira Perića-Valtera 4, 21000 Novi Sad, Serbia
{vesinboban, aklasnja} @yahoo.com
[2]Faculty of Science, Department of Mathematics and Informatics, Novi Sad
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
{mira, zjb}@dmi.uns.ac.rs

Abstract. The aim of Semantic Web is to provide distributed information with well-defined meaning, understandable for humans as well as machines. E-learning is an important domain which can be benefited from the Semantic Web technology. Ontologies, as a building structure of Semantic Web, will fundamentally change the way in which e-learning systems are constructed. The explicit conceptualization of system components in a form of ontology facilitates knowledge sharing, knowledge reuse, communication and collaboration among system components, and construction of intensive and expressive systems. In previous research, we implemented tutoring system named Protus (PRogramming TUtoring System) that is used for learning basic concepts of Java programming language. Protus uses principles of learner style identification and content recommendation for course personalization. The new version of the system called Protus 2.0, supported by several ontologies, as well as examples of its usage for performing personalization are presented in this paper. Architecture of new system extends the usage of Semantic Web concepts, where the representation of each Protus 2.0 component is made by a specific ontology, making possible a clear separation of the tutoring system components and explicit communication among them.

Keywords. Semantic Web, ontology, tutoring system, recommendation systems, personalization

1. Introduction

Semantic Web technologies seem to be a promising technological foundation for the next generation of e-learning systems [9]. *Ontology*, generally defined as a representation of a shared conceptualization of a particular domain, is a major component of the Semantic Web. The initial work on implementing ontologies as the backbone of e-learning systems is presented in [25]. Since that time, many authors have proposed the usage of ontologies in different

aspects of e-learning, such as adaptive hypermedia, personalization, and learner modelling [19].

Interest in ontologies has also grown as researchers and system developers have become more interested in reusing and sharing knowledge across systems [34]. Currently, one key obstacle to sharing knowledge is that different systems use different concepts and terms for describing domains [2]. If we could develop ontologies that might be used as the basis for multiple systems, they would facilitate sharing, reuse and common terminology.

In our previous work, we presented a general tutoring system named Protus (PRogramming TUtoring System) that is used and tested for learning basic concepts of Java programming language [40]. Personalization in Protus is based on principles of learning style identification, content recommendation and navigational sequencing [20], [41]. Learners with different learning styles have different sets of navigation sequence. Hence, learners were clustered based on their learning styles and then behavioural patterns were discovered for each learner by AprioriAll algorithm [22]. Two learners are said to be similar to each other if they are evaluated by the system with the same ratings for a similar navigational sequence. Recommendation process can be carried out according to these learning sequences based on the collaborative filtering (CF) approach [22]. The main objective of this paper is to present the improved version of the system, called Protus 2.0, that implements new, ontology based architecture of the system. This architecture will implement same recommendation techniques for performing personalization that were part of the previous version of the system, but with benefits of Semantic web technologies.

The major goal of learning systems is to support an intended pedagogical strategy [8]. In this scope, pedagogical ontologies can be associated with reasoning mechanisms and rules to enforce a personalization strategy. Often this strategy consists of selecting or computing a specific navigation sequence among the learning resources. Thus, formal semantics are required in this case to enable such computation.

Ontologies in Protus 2.0 are written in OWL [27]. To support the development of the ontologies and the translation in OWL, we use the open-source tool Protégé 4.1 [28]. The use of reasoning mechanisms and rules for knowledge representation and inference engines for reasoning are presented in our previous work [42].

The rest of the paper is organized as follows. In the second section, appropriate related work is analysed and discussed. In Section 3 the representation of components according to Semantic Web technologies within Protus 2.0 will be presented. In addition, we discuss use of ontologies in recommendation process in Protus 2.0. Performed personalization and effects of implemented ontologies are presented in Section 4. Section 5 concludes the paper and indicates directions of possible further research.

2. Related Work

Recently many researches have been focused on applying Semantic Web technologies to different aspects of e-learning [19]. Most of the developed systems use ontologies only for representation of concepts, knowledge or learners' data [13], [14].

The prototype system named SMARTIES is a totally ontology-aware system, which fully utilizes the qualities of ontology, computationally, as well as conceptually [26]. In this moment, that ontology focuses only on the abstract design of learning contents and has not been yet related to domain knowledge or learning objects to concretize the abstract design. We are going to further enhance this work by not only adapting the content modelling but also showing a way how to link semantics and content with implementing rule-based reasoning for adaptation process in Protus 2.0.

The Personal Reader [10] brings another important result in the e-learning field. This system also uses the Semantic Web to personalize and enrich e-learning contents. It presents a service architecture relying on RDF (*Resource Description Framework*) and ontologies to exchange information about learning resources, the domain, and learners. Architecture for personalized e-learning based on Semantic Web technologies was proposed in [15]. The authors propose usage of several ontologies for building adaptive educational hypermedia systems. Unfortunately, in this system ontologies do not represent the teaching strategy's functionality of a resource. A teaching strategy ontology has been implemented in the Protus 2.0 system for clear separation of performed activities in recommendation process.

Our research is closely related to the essences of QBLS system [8]. It is a web-based intelligent learning system that completely relies on Semantic Web technologies and standards. It reuses a large set of learning resources taken from the web and has been used as an online support system for lab sessions of Java programming course. On the other hand, Protus 2.0 implements a complete Java course rather than collecting unstructured learning resources from the web.

Other proposals of ontologies and their usage for several aspects of the e-learning systems, such as learner model and preferences, domain ontology, task ontology, and others, can be found in [1], [23], [32]. In the structure of these systems, the usage of ontologies focuses mainly on learning objects and their related aspects. Besides, that does not facilitate the definition and communication between the other components of the system's architecture. All elements of Protus 2.0 architecture are implemented as Semantic Web components.

In the above-mentioned papers, ontology architecture is presented but specific examples of personalization activities are omitted. The architecture for a tutoring system supported by several ontologies is presented in this paper. The implemented ontology-based architecture offers acceptable solution for the mentioned problems. Personalization actions are presented through concrete examples later in this paper.

In the structure of previously mentioned systems, the use of ontology focuses mainly on learning objects and their related aspects. Besides, that does not facilitate the definition and communication between the other components of the system's architecture. Architecture for tutoring system supported by several ontologies is described in this paper as a way of addressing and offering acceptable solution for the mentioned problems.

3. Protus 2.0 Architecture

Protus 2.0 is a tutoring system designed to support learning processes in different courses and domains. In spite of the fact that this system is designed as a general tutoring system for different programming languages, the first completely implemented and tested version was used for learning in the context of an introductory Java programming course [21]. It is an interactive system with primary goal to allow learners to use teaching material prepared within an appropriate introductory programming course but also includes a part for testing learner's acquired knowledge.

During the last decade, increasing attention has been focused on ontologies and their usage in applications related to areas such as knowledge management, intelligent information integration, education and so on. Ontology engineering, as a set of tasks related to the development of ontologies for a particular domain, offers a direction towards solving a wide range of problems brought by semantic obstacles. According to that, ontology engineering could be a key aspect for improvements of our already developed in traditional manner, tutoring system.

Ontologies allow specifying formally and explicitly the concepts, their properties and relationships [13]. Educational ontologies such as: for presenting a domain (*domain ontology*), building learner model (*learner model ontology*), presenting of activities in the system (*task ontology*), specifying pedagogical actions and behaviours (*teaching strategy ontology*), defining the semantics of messages sent among components (*communication ontology*) and specifying behaviours and techniques at the learner interface level (*interface ontology*), must be included in the system [9]. A repository of ontologies must be built to achieve easier knowledge sharing and reuse, more effective learner modelling and easier extension of whole system. Ontologies are structured following the SCORM (*Sharable Content Object Reference Model*) e-learning standard [30] using the formal ontology language OWL [32]. This ontological representation enables not only to represent meta-data but also reasoning in order to provide the best solution for each individual learner [42]. General ideas, of redefined ontology-based architecture for Protus 2.0 have been presented in [20]. This architecture is based on experiences gained from similar web-based learning systems [5], [24], [31] and architecture for ontology-supported adaptive web-based education systems suggested in [7], [25], [37]. All the proposed architectures are highly modular with four central components: the application module, the adaptation module,

the learner model and domain module. In all proposed architectures, the adaptation module is explicitly separated from the domain module, but another component is introduced in Protus 2.0 as in [25] – the application module. This module is used for storing adaptation rules used in further personalization process. Figure 1 depicts the general architecture of the redesigned and extended version of Protus 2.0 system.

It is important to note that the original architecture of Protus did not bring any kind of homogenous representation of components. Each one was represented by different formats, using a variety of tools. The purpose, of our current research activities, was to represent each component of the system in form of the ontology. According to that, the level of abstraction of this architecture will be higher [18].

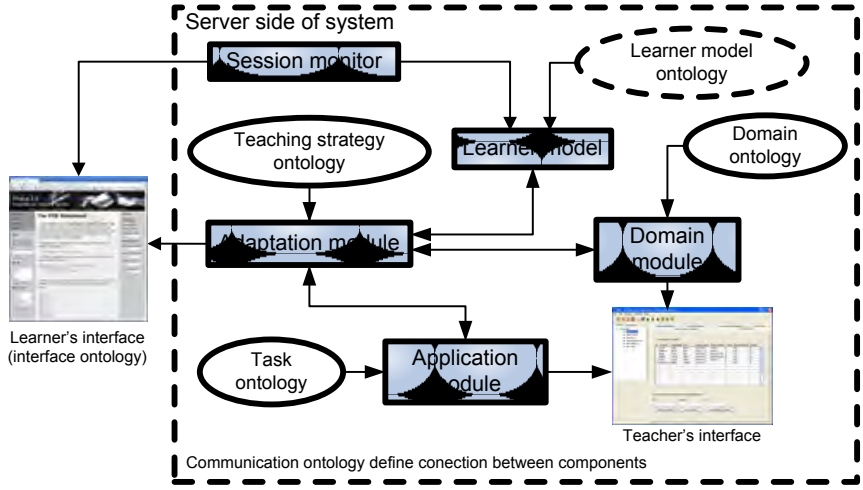


Fig. 1. Protus 2.0 Architecture

This approach will make it easier to understand the role of each component and, consequently, to promote interoperability among the components of the architecture. The developed system is modular, which allows better flexibility and future replacement of various components as long as they comply with the current interface. In the rest of the section, details of construction of different kinds of ontologies and their use for performing personalization employed in our system will be separately addressed.

Concrete examples of the ontologies and their use for the purpose of personalization within Java tutorial will be presented.

3.1. Domain Ontology

One of the main goals of the learning process is to understand and to acquire a body of knowledge for a given domain [29]. Domain model presents storage for all essential learning material, tutorials and tests. It describes how the

content intended for learning has to be structured [3]. Often the domain model can be structured as a taxonomy of concepts, with attributes and relations connecting them with other concepts, which naturally leads to the idea of using ontologies to represent this knowledge.

The complete Java course contains several concepts (lessons) [16]. Therefore, the Java course in Protus 2.0 contains: an introductory lesson, syntax, loop statements, execution control, etc. (Figure 2). To each concept any number of different resources (text files, images, animations, etc.) can be assigned. All resources are assigned depending on their *Resource type*: theory, examples, assignments, exercises, syntax rules, and so on.

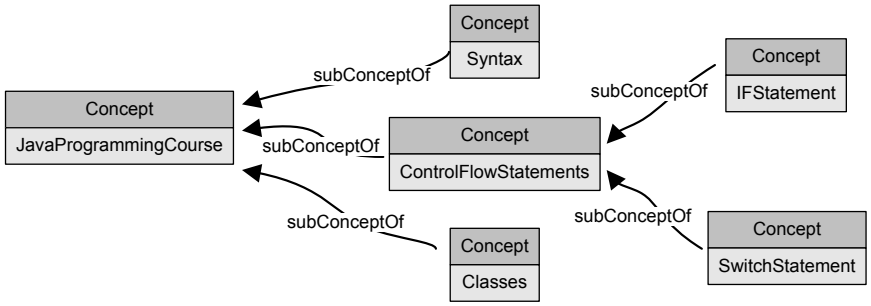


Fig. 2. An excerpt of ontology as domain topology of Protus 2.0

Concepts. In Figure 2, an excerpt of a domain ontology covering basics of Java programming concepts with *subConceptOf* relationships between these concepts has been shown. This figure depicts the root concept with some of its sub concepts: *Syntax*, *ControlFlowStatements* and *Classes*. The *ControlFlowStatements* concept is further specialized and fine-grained into *IFStatement* and Switch*Statement*. Clear specification of other relations between concepts will be useful for further personalization purposes.

An example of instance of *Concept* class that is used to collect information about the *For Statement* concept is presented in Table 1.

Table 1. Example of instance of *Concept* class

Property description	Property name	Property value	Property type
Concept's id	hasId	C009	Datatype property
Concept's name	hasName	ForStatement	Datatype property
Resource's type	hasResource	R017	Object property
Superclass	subConceptOf	LoopStatements	Object property
Prerequisite	hasPrerequisite	ExecutionControl	Object property
Prerequisite	hasPrerequisite	Syntax	Object property

A *Property* is a directed binary relation that specifies class characteristics. They represent attributes of instances and sometimes act as data values (*Datatype property*) or link to other instances (*Object property*). OWL also has a third type of property - *Annotation properties*. Annotation properties can be used to add information (metadata) to classes, individuals and object/datatype properties. OWL allows classes, properties, individuals and the ontology itself to be annotated with various pieces of information/meta-data. These pieces of information may take the form of auditing or editorial information. For example, it could be details about creation date, author, comments or references to resources such as web pages etc.

This particular instance of *Concept* class (Table 1.) has unique id: C009. It has been used for defining a lesson named *ForStatement* and it contains data about its superclass (it is *subConcept* of *loopStatements* concept) and concepts that are prerequisite for it (*ExecutionControl* and *Syntax*).

Resources. All concepts must be supported with various types of resources. When the learner accesses a course, Protus 2.0 can infer which resources could be suitable for presentation to the learner.

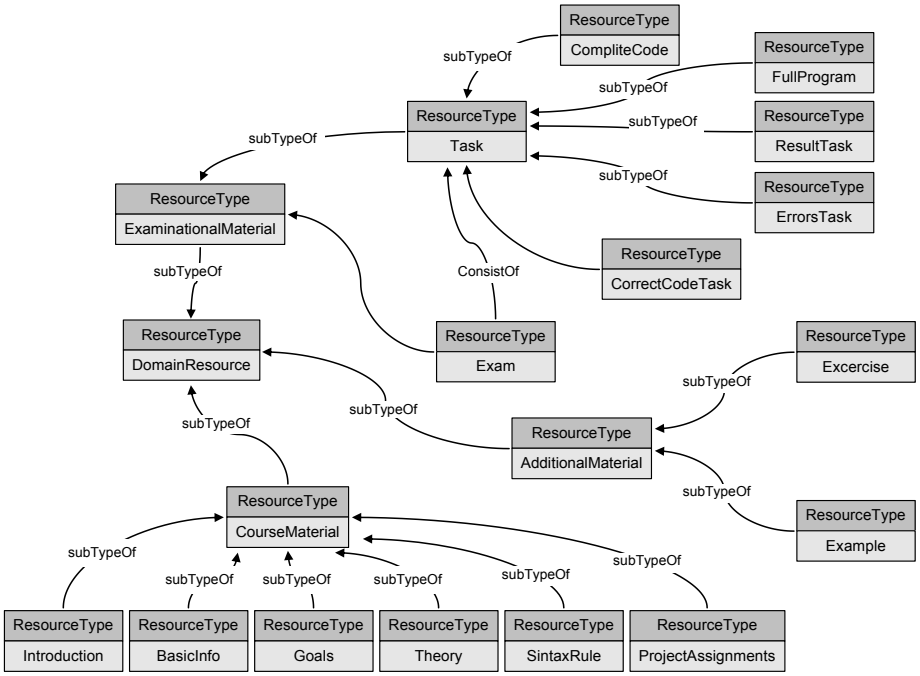


Fig. 3. An excerpt of ontology as resource topology of Protus 2.0

An excerpt of ontology as *Resource type* topology is depicted in Figure 3. The ontology depicts resource types in the programming domain. The most general resource type is *DomainResource*. *DomainResource* has three subtypes: *CourseMaterial, AdditionalMaterial* and *ExaminationMaterial*.

Classes *CourseMaterial* and *AdditionalMaterial* represent the theoretical and practical explanations, respectively, that are displayed to the learners. *ExaminationMaterial* can be further specialized to *Task* and *Exam*. The *Exam* is consisted of various *Tasks* [17]. Tasks could include code completion, code correction, listing errors, etc.

CourseMaterial can be further specialized into *Introduction*, *BasicInfo*, *Goals*, *Theory*, *SintaxRule* and *ProjectAssignements,* which corresponds to the essential elements of a programming language course.

Ontology presented in the Figure 3 further provides information for the *Task ontology* and the *Teaching strategy ontology* which will be explained in more details in the rest of the section.

Details about resources are kept in *Resource* class instances. Each instance of the *Resource* class contains basic information on individual resources, which will later be used for the subsequent selection of appropriate resources in the process of personalization. Specific type and role are determined for every resource.

An example of instance of the *Resource* class that is used to display the syntax rules of *for* statement is presented in Table 2.

Table 2. Example of instance of the resource class

Property description	Property name	Property value	Property type
Resource's id	hasId	R017	Datatype property
Resource's name	hasName	forLoop017	Datatype property
Resource's type	isTypeOf	Syntax rule	Object property
Concept's type	isResourceFor	For Statement	Object property
Resource's role	supports	Visual style	Datatype property
Is resource visited?	isVisited	yes	Datatype property
Is resource recommended	isRecommended	no	Datatype property
File Type	hasFileType	jpg	Datatype property
Concept's role	hasRole	definition	Datatype property
Link to used figure	hasFigure	Figure6.jpg	Annotation properties

This particular instance of *Resource* class has unique id: R017. It is used for presenting a *syntax rule* for a lesson (concept) named *ForStatement* and it contains a link to a certain jpg file (Figure 4) that will be presented to the learner if the system chooses this resource during personalization activities.
All resources are grouped by their type, role and the concept they support and that present a basis for successful recommendation during the personalization process.

3.2. Task Ontology

Task ontology is a system of vocabulary for describing problem-solving structure of all existing types of tasks domain independently. It complements the domain ontology by representing semantic features of the problem solving [9]. Task ontology specifies domain knowledge by giving roles to each object and relations between them. For instance, what role a paragraph in a textbook, can play.

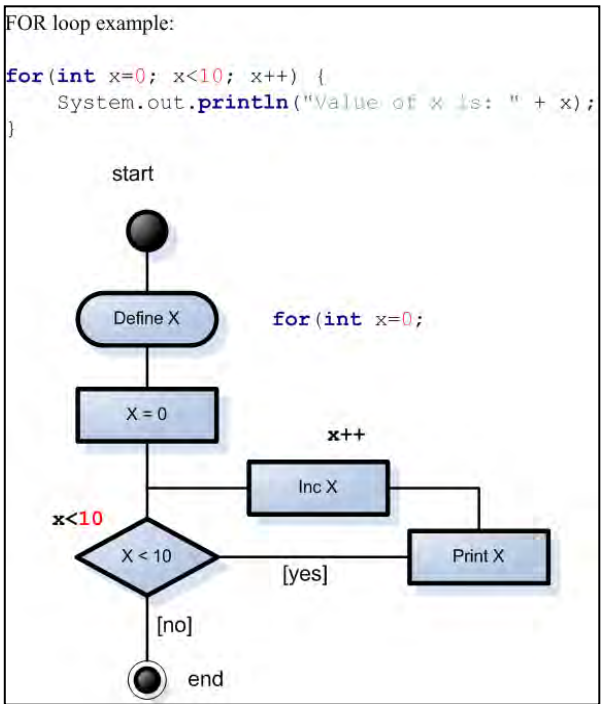


Fig. 4. Figure resource

Task ontology shows the role of a specific resource from the domain ontology. For example, if a resource has *fact* or *definition* role it is used to increase basic knowledge and if its role is *example*, then it is used to increase learner‘s practical skills.

An excerpt of task ontology of resources in Protus 2.0 is depicted in Figure 5. The ontology represents learning material grouped by the resources. The class *Concept* is used to annotate a unit of knowledge, which is represented by some *Resource*.

Like in [8], concepts and resources are related by the *hasResource* property. Concepts can be arranged by the *hasPrerequisite* property. The *hasPrerequisite* property is proposed for navigational purposes. It allows pointing out concepts that must be known before starting to study a concept,

and the concepts for which it is a prerequisite. Concept will not be covered unless that the prerequisite condition is satisfied. There can be a different sequence of resources that depends on the navigational sequence determined for a particular learner.

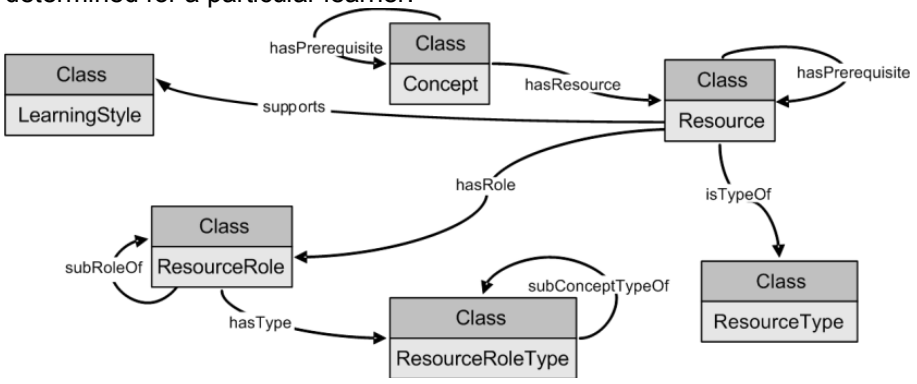


Fig. 5. An excerpt of task ontology of resources

Resources play certain roles in particular concept fragments. For example, some resources represent the crucial information, while the others just represent a mean to provide additional information or a comparison. In the proposed ontology, we represent these facts by instances of the *ResourceRole* class and its two properties: *hasRole* and *supports*. For example, resources like *BasicInfo* and *Example* have different roles. The role of the first is to represent introductory information for lesson and the role of the former is to provide additional information. On the other hand, both concepts support adaptation to learner with *Reflective* style of learning [12]. Resource properties can be further extended by assigning a *ResouceType*. Similarly, the resources roles can be further extended by specifying their types. Concepts, their types and resources form the task ontology of Protus 2.0 system.

3.3. Learner Model Ontology

The learner model stores personal preferences and information about the learner's mastery of domain concepts [36]. The information is regularly updated according to the learner's interactions with the content and is used by the *Teaching strategy ontology* to draw conclusions and decisions. This ontology (Figure 6.) offers the opportunity to map all information about the learner, from confidential data, like password, to a knowledge evolution history.

The class *Learner* is built from three components: *Performance*, *PersonalInfo*, and *LearningStyle*. These three classes are related to association through *hasPerformance*, *hasInfo*, and *hasLearningStyle* properties.

Class *LearningStyle* represents the preferred learning style for particular learner. This class offers four categories to the dimensions of the Felder-Silverman Learning Style Model (sequential/global, active/reflective, visual/verbal and sensing/intuitive) [12].

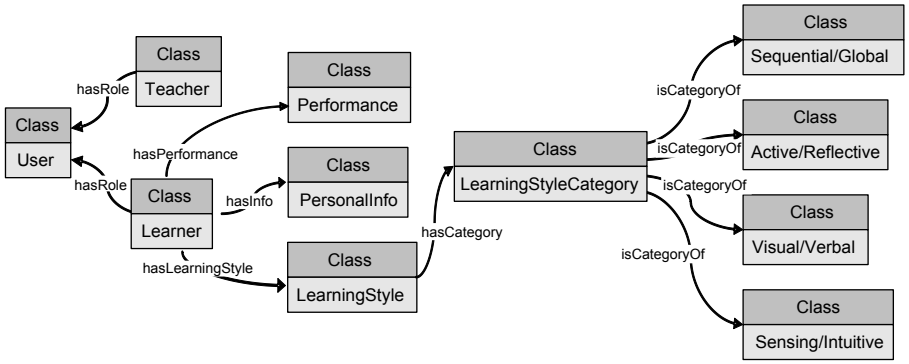


Fig. 6. Learner model ontology of Protus 2.0

During a learning session, the learner interacts with a tutoring system. Learner interactions can be used to draw conclusions about his/her possible interests, goals, tasks, knowledge, etc. These conclusions can be used later for providing personalization. Ontology for learner observations should therefore provide a structure of information about possible learner interaction. Figure 7 depicts such ontology as a part of *Learner model ontology*. Learner performance is maintained according to a class *Interaction*. *Interaction* is based on actions taken by a specific learner, during a specific *Session*. Interaction implies a *Concept* learned from the experience, which is represented by the *conceptUsed* property. *Interaction* has a certain value for *Performance*, which is in this context defined as a floating-point number and restricted to the interval from 1 to 5. This ontology is responsible for updating the *Learner model ontology*.

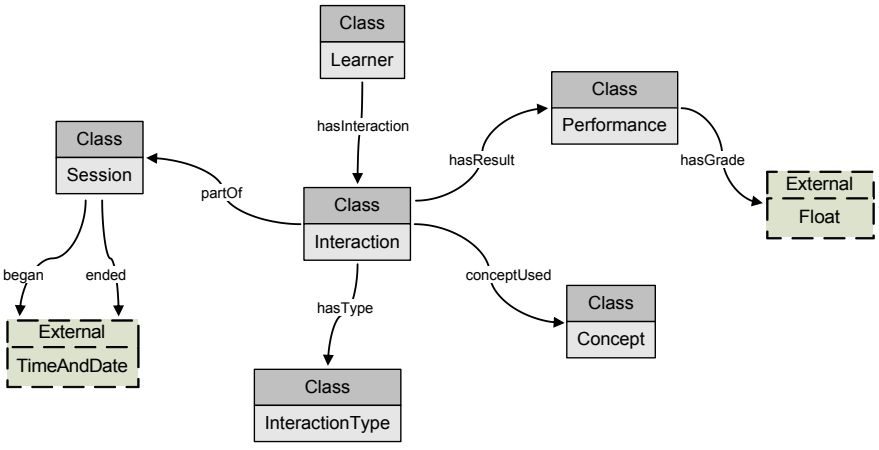


Fig. 7. Ontology for learner observation and modelling

Class
Concept
External
Float

An example of instance of *Interaction* class that is used to keep track of the learner's interaction during one session is presented in Table 3.

Table 3. Example of instance of *Interaction* class

Property description	Property name	Property value	Property Type
Interaction's id	hasId	I007	Datatype property
Session	partOf	S1012	Object property
Interaction's type	hasType	T02 (test)	Object property
Used concept	conceptUsed	C032	Object property
Learner	whoInteracted	L01	Object property
Performance	hasResult	P01	Object property

This particular instance of *Interaction* class has unique id: I007. It is formed during session S1012 when learner with id L01 took test and gained results, which are all collected in instance of *Performance* class with id P01. Instances of class *Performance* contain, among other, data about grade that learner earned during current testing. Based on that *Performance* data, system makes decision in further personalization within *Teaching strategy ontology* described in next section.

3.4. Teaching Strategy Ontology

Authoring of adaptation and personalization is actually authoring of learner models and applying different adaptation strategies and techniques to ensure efficient tailoring of the learning content to the individual learners and their learning styles and navigation sequencing [1].

Figure 8. shows how the adaptation is carried out by the *Teaching strategy ontology*. The decisions are drawn on the basis of the information contained in the *Condition* class (that is generated by the information about learning style and performance of the learner) as well as teaching goals and previous behaviour patterns. These conditions are composed of data coming from several other components such as *Learner model ontology*, *Task ontology* and *Domain ontology*.

Personalization presents the choice of the most appropriate learning pattern or resource that will be recommended to the learner. This action depends of many conditions but it implies only one decision. The decision determines what concept and resource the system is going to present for the learner.

An instance of the *Condition* class that was formed based on the *Performance* data of every learner is presented in Table 4. This particular

instance of *Condition* class has unique id I006. It contains data collected based on learner's learning style and performance.

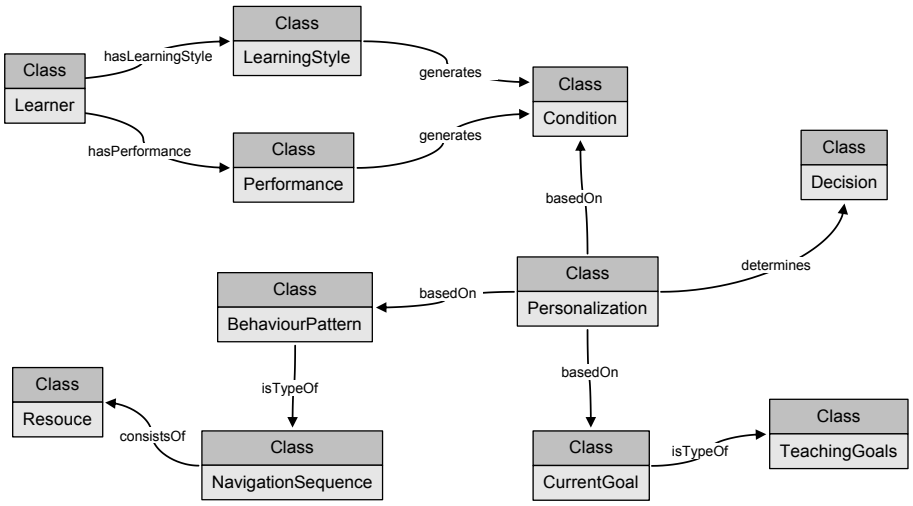


Fig. 8. Teaching Strategy ontology of Protus 2.0

An instance of the *BehaviourPattern* class presents specific type of *NavigationalSequence* class. Former consists of series of resources that learner had interacted with.

Table 4. Example of instance of *Condition* class

Property description	Property Name	Property value	Property type
Condition's id	hasId	I006	Datatype property
Learning style of learner	generatedBy	LS03	Object property
Learning style Category	hasLearningStyleCategory	visual	Datatype property
Learning style domain	hasLearningStyleDomain	Information Reception	Datatype property
Learner's performance	generatedBy	P01	Object property
Personalization	Generates	PR09	Object property

For example, an instance of the class *BehaviourPattern* contains the information presented in Table 5.

This particular instance of *BehaviourPattern* class is type of Navigational sequence marked as NS02, that learner made during session S1012, with

rating of 0.37 and it generates instance of *Personalization* class marked as PR09.

All personalization activities within Protus 2.0 are performed based on previously mentioned data in instances of *Condition* and *BehaviourPattern* data. Section 4 will present a few examples of performed personalization based on all collected data about learner's interaction with the system.

Table 5. Example of instance of *BehaviourPattern* class

Property description	Property name	Property value	Property type
BehaviourPattern's id	hasId	BP0016	Datatype property
Navigational Sequence	isTypeOf	NS02	Object property
Session	partOf	S1012	Datatype property
Personalization	generate	PR09	Object property
Ratings of navigational sequence	hasRate	0,37	Datatype property

4. Personalization in Protus 2.0

Protus 2.0 system offers two types of personalization to each individual learner, personalization based on learning styles of learners and personalization based on mining the frequent sequencing.

When learners start their interaction with Protus 2.0, the first step is to cluster learners based on their learning style. Behavioural patterns are discovered for each learner by AprioriAll algorithm, based on chosen options during learning (details about performed algorithm in Protus are presented in [20]). Recommendation list of material has to be presented to learner is created according to the data from the *Learner model ontology* and ratings of the frequent sequences, provided by the Protus 2.0 system. All recommendation actions are done in real-time during learning sessions. The provided recommendation is expected to have a higher accuracy in matching learners' requirements to learning material and thus a higher level of acceptance by the learners.

Queries over ontology data has been performed using Sparql – query language for RDF [33] and Semantic Web rule language - SWRL has been used for rule-based reasoning [35]. Details about implemented adaptation rules in Protus 2.0 are presented in [38], [39]

4.1. Learner's interaction with Protus 2.0

In this section, we will explain recommendation procedures for a new learner. The new learner signs up by using the registration form in order to create an initial personal profile and update the *Learning model ontology*. Each profile stores personal information supplied directly by the learner, i.e.: last name, first name, login, previous knowledge, preferences, etc. (known as static information), and information about learning style, current progress, and behaviour (known as dynamic information). When learners are already registered to the system, their learning styles need to be tested. The learner has to fill-in a questionnaire that is used to determine his/her preferred learning style [21]. The learning style of the student indicates a preference for some presentation methods over others. These results are stored in the *Learner model ontology* (Figure 6), which will be used for the initial adaptation in Protus 2.0.

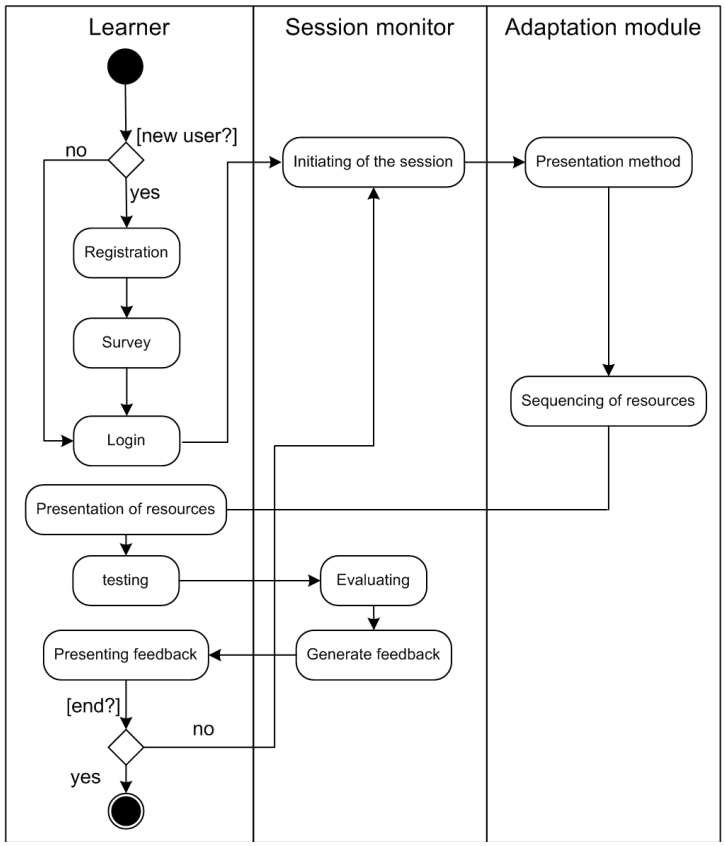


Fig. 9. Learning process in Protus 2.0

When a learner (nevertheless, he/she is new or the old one) is logged in, a session is initiated based on the learner's specific learning style already

stored in *Learning model ontology*, and sequence of lessons is recommended to him/her (Figure 9). The learner can freely change the order of lessons he/she is attending. After selecting a lesson, from the collection of lessons available in Protus 2.0, the system chooses a presentation method for the lesson based on the learners' preferred learning style. For the rest of the lesson, learners can freely switch among presentation methods using the media experience bar. That option was provided because initial assessment of the user's learning styles can be misleading in overall process of personalisation. When the learner completes the sequence of learning materials, the system evaluates the learner's knowledge degree for each lesson. The test contains several multiple-choice questions and code completion tasks. Protus 2.0 then provides feedback to the learner on his/her answers and gives the correct solutions after the learner finished the test.

Recommendations cannot be made for the whole pool of learners, because even for learners with similar learning interests, their ability to solve a task can vary due to variations in their knowledge level. In this approach, we perform a data clustering technique as a first step to cluster learners based on their learning styles. These clusters are used to identify coherent choices in frequent sequences of learning activities. Then, a recommendation list can be created according to the ratings of these frequent sequences provided by the Protus 2.0 system. The details of the whole process are presented in the rest of the paper.

4.2. Learning styles

It is obvious that different learners have different preferences, needs and approaches to learning. Psychologists call these individual differences learning styles. Therefore, it is very important to accommodate for the different styles of learners through learning environments that they prefer and find more efficient. Learning styles can be defined as unique manners in which learners begin to concentrate on, process, absorb, and retain new and difficult information [11].

There are over seventy identifiable approaches to investigate and/or describe learning style preferences. We used one such data collection instrument, called Index of Learning Styles (ILS) [12]. The ILS is a 44-question, freely available, multiple-choice learning styles instrument, which assesses variations in individual learning style preferences across four dimensions or domains [22]. These are *Information Processing*, *Information Reception*, *Information Perception* and *Information Understanding*. Within each of the four domains of the ILS there are two categories:

- *Information Processing*: Active and Reflective learners,
- *Information Perception*: Sensing and Intuitive learners,
- *Information Reception*: Visual and Verbal learners,
- *Information Understanding*: Sequential and Global learners.

The preferred learning style can be investigated by offering the learner a free choice between an example, an activity or an explanation at first, and by observing a pattern in the choices, he/she makes [22].

At the beginning of the session Protus 2.0 requests information about the status of the course from the *Learner model ontology* for the particular learner (Figure 10). This data includes information about the current lesson and the learning style category of the learner within one of the four domains of the ILS (Figure 6). Request for appropriate resources which will be presented to the learner, based on this data, is sent to the *Application module*. Further, all activities of learners are monitored, as well as all requests he/she send to the system.

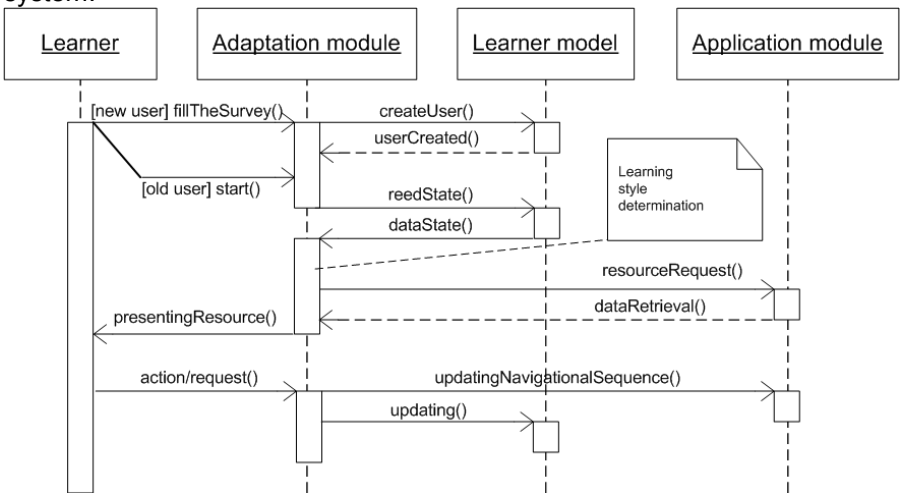


Fig. 10. Adaptation based on the learning styles

If the learner successfully learned a specific concept (Table 1), which is supported with a specific resource (Table 2), then the system should add that resource and details about performance to the successful navigational pattern (Figure 8). In addition, if the learner interacts with a specific resource and during that interaction he/she took the test and earned a specific grade, then the system should memorize that learner's performance (Table 3) and mark that resource as learned (Table 2).

If the learner does not provide the required level of performance results within a session based on the presentation method used for his/her certain learning style category, his/her current learning style category will be modified. In those cases, the system changes the current learning style of the learner to its alternative, from the same domain. Learning styles are grouped in pairs (active and reflective, sensing and intuitive, visual and verbal, sequential and global), therefore every learning style has only one alternative within the domain.

For example, if a learner with *Verbal* learning style interacts with the system and during that interaction he/she had accessed appropriate concept but not earned sufficient grade (required grade level is kept in global value

required), then, the learning style of that learner should be changed to its alternative from Information reception domain: *Visual* learning style. That implies that in the next session, the learner will be presented with resources that are defined to support a particular learning style category.

4.3. Navigational patterns

Resource sequencing is a well-established technology in the field of application of intelligent tutoring systems in educational processes [4]. The idea of resource sequencing is to generate a personalized course for each learner by dynamically selecting the most optimal teaching actions, presentation, examples, task or problems at any given moment. By optimal teaching action, it is considered an operation that in the context of other available operations brings the learner closest to the ultimate learning goal. Most often, the goal is to learn and acquire some knowledge up to a specific level in an optimal amount of time. Learners could follow different paths based on their preferences and generate a variety of learning activities. All these variations in series of learning activities are recorded by the Protus 2.0 system.

In order to monitor learner's performance during the session, Protus 2.0 records results of learner's interaction, earned grades and data about used concepts (navigation through resources). These results are used in building a global database of navigational patterns.

When the learner completes the sequence of learning materials, the Protus 2.0 system evaluates the learner's acquired knowledge [21]. The learners' grades can be interpreted according to the percentage of correct answers. Two learners are said to be similar to each other if they are evaluated by the system with the same ratings for a similar navigational sequence. Ratings of frequent sequences are not calculated only by followed sequences itself but earned grades throughout session are also included in calculation. Therefore, every system-imposed path still counts towards placing the learner in a particular cluster. Recommendation process can be carried out according to these learning sequences based on the collaborative filtering approach that is described in our previous work [22], [40]. Here, one practical example will be presented.

After each request or action of learner, the system examines the current resource sequence and makes comparison with navigational patterns of previous users (Figure 11). Protus 2.0 finds similar users and creates a recommendation list according to the ratings of the frequent sequences.

For example if a specific navigational sequence of resources has been recommended to the learner, then the system should recommend to him/her the next specific resource that belongs to that navigational sequence. Recommendation status of that resource is set to true (Table 2), therefore one of the several changes in user interface must be made (depending on the included resource type):

- link to that resource is annotated or highlighted (Figure 12a).

- the interface elements for sequential navigation will be hidden, giving the learner possibility to freely jump through the courseware (in a case of sequential learning style category) or presented (in a case of sequential learning style category, Figure 12b).
- additional tabbed pane elements will be added to related or more complex content to help situate the learnt subject and contribute in creating clear overall view on the subject being thought (Figure 12c).

Details of SWRL rules that retrieve data from Protus 2.0 ontology and make appropriate decisions for performed adaptation are presented in [38], [39].

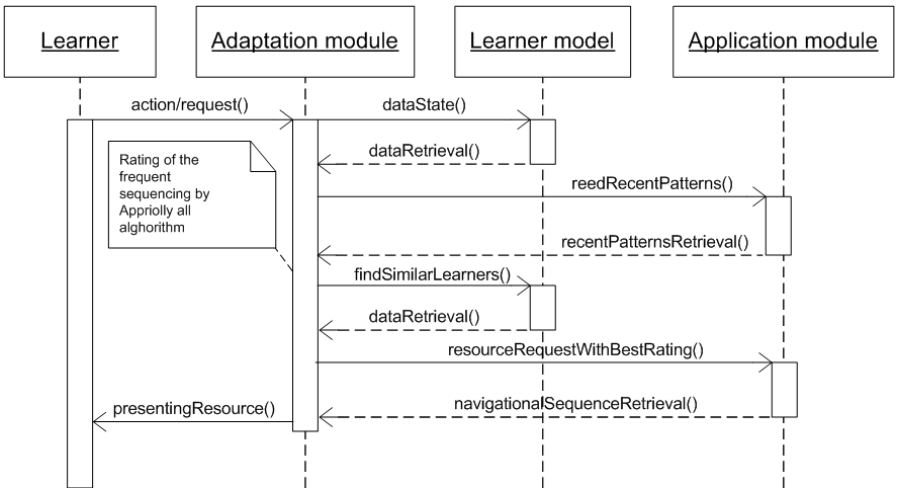


Fig. 11. Adaptation based on the navigation pattern

a. Annotation b. Presented interface elements c. Adding elements

Fig. 12. User interface adaptation

4.4. Final Remarks

The architecture for a tutoring system supported by several ontologies was presented as a way of addressing the problems of maintenance and reuse of components of the learning system. The implemented architecture extends

the use of ontology, where the representation of each component is made by a specific ontology. This way, it is possible to:

- promote a clear separation of concerns of the tutoring system components
- make explicit the communication among the components
- specify the interface for easy updating of the components
- emphasize the gains of the use of Semantic Web in the development of tutoring systems.

Ontology-based architecture also provides better interoperability and reusability of systems components in the future that will allow implementing additional programming courses as well as courses from other domains.

Preparation stage for building ontology-based architecture of Protus 2.0 utilized the main disciplines of knowledge engineering [6]. The four preparation steps were:

1. developing an ontological knowledge model by gathering expertise related to learning objects and content sequencing
2. representing the domain model using OWL ontology
3. establishing individual relationships and adaptation rules using SWRL and
4. asserting factual knowledge based on the ontology schema.

The above processes provide the basis for building an initial ontological knowledge base. Both description logic reasoning and the rules engine were further applied to extend the inference power to establish a runtime knowledge base. In our previous works, we presented SWRL rules for altering pattern navigation [38, 42] and rules for building learner model [39, 42] in Protus 2.0.

These ontologies may serve as a foundation that can be extended and modified to present the ontology for other adaptive systems.

The use of ontologies to model the knowledge in Protus 2.0 system represents a key aspect for the integration of various information that are presented to the learner, for supporting collaboration within learning community, for improving information retrieval, and more generally, it is important for reasoning on available knowledge. In Protus 2.0, ontologies are used to model educational domains and to build, organize and update specific learning resources (i.e. learning objects, learner profiles, learning paths, etc.). Ontology represents the terminology of a programming domain, defining its essential knowledge. Ontologies in Protus 2.0 will also be used to support semantic search, making possible to query multiple repositories and discover associations between learning objects that are not directly understandable.

This Semantic web approach allows implementation of adaptation customized to different requirements. The learner demand is derived from the knowledge contained in the ontology. Various conditions in Protus 2.1 are captured in the body of SWRL rules that are described in [42]. As a result of the firing of rules, recommendations in the form of various content presentations are generated, which can be used to implement the concept of adapted content and adapted navigation.

Practically, Protus 2.0 do not offer any kind of improvements regarded to personalization process. This process is the same as in the previous version of Protus. Architecture supported by the Semantic Web Technologies has been proposed in this paper in order to implement new, ontology based architecture, while evaluation of the performed personalization process has been described in detail in [42]. Learners were basically satisfied with previous version, therefore similar results are expected with this new version of the system. Protus 2.0 is in the process of implementation and results of the use of the system in the classroom will be the subject of our future work.

5. Conclusion

In this paper we proposed how Semantic Web technologies and in particular ontologies can be used for improving functionalities of an existing Java tutoring system. The architecture for such an adaptive and personalized tutoring system that completely relies on Semantic Web standards and technologies has been presented. The form of several ontologies has been proposed which correspond to the components of a tutoring system.

Ontologies will fundamentally change the way in which systems are constructed. Today, knowledge bases are still built with little sharing or reusing - almost each one is built from scratch. In the future, intelligent systems developers will have libraries of ontologies at their disposals. Rather than building from scratch, they will assemble knowledge bases from the libraries. This should greatly decrease development time while improving the robustness and reliability of the resulting knowledge bases.

The explicit conceptualization of system components in a form of ontologies facilitates knowledge sharing, knowledge reusing, communication and collaboration among system components, and construction of intensive and expressive systems. Ontologies are being more and more widely used for constructing systems that require an explicitly encoded knowledge. Therefore, those systems are able to interoperate better providing learners with an extended access to information resources.

The proposed architecture is modular, which allows higher flexibility and future replacement of various components as long as they comply with the current interface.

Previous version of Protus system was intensively tested and several experiments were carried out in order to show suitability of using recommendation technics for suggesting online learning activities to learners based on their learning style, knowledge and preferences. The plan is to incorporate Protus 2.0 in traditional programming courses. With this new Semantic web, supported tutoring system higher scalability and easier maintenance of the system are expected.

Although ontologies have a set of basic implicit reasoning mechanisms derived from the description logic, which they are typically based on (such as classification, instance checking, etc.), they need rules to make further

inferences and to express relations that cannot be represented by ontological reasoning. Thus, ontologies require a rule system to derive/use further information that cannot be captured by them, and rule systems require ontologies in order to have a shared definition of the concepts and relations mentioned in the rules. For the future work, we plan to present complete set of personalization rules that will allow reasoning on the instances of ontologies.

Acknowledgments. This paper is part of the research project Infrastructure for Technology enhanced Learning in Serbia supported by the Ministry of Science, Technologies, and Development of the Republic of Serbia [Project No. III47003].

References

1. Aroyo, L., Mizoguchi, R.: Process-aware Authoring of Web-based Educational Systems. In Proceedings of the International Workshop on Semantic Web and Web-based Education SWWL, Velden, Austria, 212-221. (2003)
2. Bodroža-Pantić, O., Matić-Kekić S., Jakovljev, B., Marković, Đ.: On MTE-model of mathematics Teaching: Studying the Problem Related to a Plane Division Using the MTE-model, *International Journal of Mathematical Education in Science and Technology* 39(2):197-213. (2008)
3. Bork, A.: Tutorial Learning for the New Century. Journal of Science Education and Technology, Vol. 10, No. 1, 57-71. (2001)
4. Brusilovsky, P., Vassileva, J.: Course sequencing techniques for large-scale webbased Education. International Journal of Continuing Engineering Education and Lifelong Learning 2003 - Vol. 13, No.1/2, 75-94. (2003)
5. Chen, C. M.: Ontology-Based Concept Map for Planning a Personalized Learning Path. British Journal of Educational Technology, Blackwell publishing, 1-31. (2008)
6. Chi, Y.L.: Ontology-based curriculum content sequencing system with semantic rules. Expert Systems with Applications Vol. 36, 7838–7847. (2009)
7. De Bra, P., Aroyo, L., Chepegin, V.: The next big thing: adaptive Web-based systems. Journal of Digital Information 5, Article No. 247. (2004)
8. Dehors, S., Faron-Zucker, C.: QBLS: A semantic Web Based Learning System. In Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications, 795-802. (2006)
9. Devedžić, V.: Semantic Web and Education. Springer Science, New York; (2006)
10. Dolog, P., Nejdl, W.: SemanticWeb Technologies for the Adaptive Web. The Adaptive Web, LNCS 4321, Springer-Verlag Berlin Heidelberg, 697–719. (2007)
11. Dunn, R., Dunn, K., Freeley, M. E.: Practical applications of the research: responding to students' learning styles-step one. Illinois State Research and Development Journal, Vol. 21, No. 1, 1–21. (1984)
12. Felder, R. M., Soloman, B.A.: Index of learning styles questionnaire. Retrieved 17 December 2009, from http://www.engr.ncsu.edu/learningstyles/ilsweb.html, (1996)
13. Gascueña, J. M., Fernández-Caballero, A., González, P.: Domain Ontology for Personalized E-Learning in Educational Systems. In Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies, 456-458. (2006)

14. Hee Lee, C., Hyun Seu, J., Evens, M. W.: Building an Ontology for CIRCSIM-Tutor. In Proceedings of the 13th Midwest AI and Cognitive Science Society Conference, Chicago, 161–168. (2002)
15. Henze, N., Dolog, P. Hejdl, W.: Reasoning and Ontologies for Personalized E-Learning in the Semantic Web. Educational Technology & Society, Vol. 7, No. 4, 82-97. (2004)
16. Ivanović, M., Bađonski, M., Budimac, M., Pešović, D., Programski jezik Java, PMF, Depatrman za matematiku i informatiku, Novi Sad, 2005
17. Ivanović, M., Pribela, I., Vesin, B., Budimac, Z.: Multifunctional Environment For E-Learning Purposes. Novi Sad Journal of Mathematics, NSJOM, Vol. 38, No. 2, 153-170. (2008)
18. Jacinto, A.S., Parente de Oliveira, J. M.: An ontology-based architecture for Intelligent Tutoring System. Interdisciplinary Studies in Computer Science Vol. 19, No. 1, 25-35. (2008)
19. Jovanović, J., Rao, R., Gašević, D., Devedžić V., Hatala, M.: Ontological Framework for Educational Feedback. In Proceedings of the SWEL Workshop of Ontologies and Semantic Web Services for IES, AIED 54-64. (2007)
20. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: Integration of Recommendations into Java Tutoring System. In Proceedings of the 4th International Conference on Information Technology ICIT 2009 Jordan, Paper no 154. (2009)
21. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: Integration of recommendations and adaptive hypermedia into Java tutoring system. Computer Science and Information Systems, Vol. 8, No. 1, 211-224. (2011)
22. Klašnja-Milićević, A., Vesin, B., Ivanović, M., Budimac, Z.: E-Learning Personalization Based on Hybrid Recommendation Strategy and Learning Style identification. Computers & Education Vol. 56, 885-899. (2011)
23. Lee, M.C., Yen Ye, D., Wang, T.I.: Java Learning Object Ontology. In Proceedings of the Fifth IEEE International Conference on Advanced Learning Technologies, 538-542. (2005)
24. Merino, P. J. M., Kloos, C.D.: An Architecture for Combining Semantic Web Tehniques with Intelligent Tutoring Systems. ITS, Springer-Verlag Berlin Heideberg, 540-550. (2008)
25. Mizoguchi, R., Bourdeau, J.: Using Ontological Engineering to Overcome AI-ED Problems. International Journal of Artificial Intelligence in Education, Vol. 11, No. 2, 107-121. (2000)
26. Mizoguchi, R., Hayashi, Y. Bourdeau, J.: Inside Theory-Aware and Standards-Compliant Authoring System. SWEL Workshop of Ontologies and Semantic Web Services for IES, AIED, 1-18. (2007)
27. OWL Web Ontology Language Reference. In: Dean, M., Schreiber, G. (eds.) W3C Recommendation, Retrieved February 10th 2004, http://www.w3.org/TR/2004/REC-owl-ref-20040210/
28. Protégé http://protege.stanford.edu/, Retrieved December 19th 2011
29. Razmerita, L., Nabeth, T., Angehrn, A., Roda, C.: InCA: An Intelligent Cognitive Agent-Based Framework For Adaptive And Interactive Learning, Cognition and Exploratory Learning in Digital Age. In Proceedings of the IADIS International Conference, Lisbon, Portugal, 373-382. (2004)
30. SCORM 2004 3rd Edition - Overview, 16 November 2006, Advanced Distributed Learning. http://www.adlnet.gov.
31. Rodríguez-González, A., Torres-Niño, J., Jimenez-Domingo, E., Gomez-Berbis, J. M., Alor-Hernandez, G.: AKNOBAS: A Knowledge-based Segmentation

Recommender System based on Intelligent Data Mining Techniques. Computer Science and Information Systems, Vol. 9, No. 2, 713-740. (2012)
32. Sosnovsky, S., Mitrovic, A., Lee, D.H., Brusilovsky, P., Yudelson, M., Brusilovsky, V.: Towards Integration of Adaptive Educational Systems: Mapping Domain Models to Ontologies. In Proceedings of the 6th International Workshop on Ontologies and Semantic Web for E-Learning at ITS 2008, Montreal, Canada, 60-64. (2008)
33. Sparql – query language for RDF, www.w3.org/TR/rdf-sparql-query/
34. Swartout, W., Tate, A.: Ontologies. Intelligent Systems and their Applications, IEEE, Vol. 14, No. 1, 18-19. (1999)
35. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Retrieved 19.12.08 from http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/
36. Ullrich, C.: Description of an Instructional Ontology and its Application in Web Services for education. In Poster Proceedings of the 3rd International Semantic Web Conference, 93-94. (2004)
37. Vesin, B., Ivanović, M.: Modern Educational Tools. In Proceedings of the PRIM2004, 16th Conference on Applied Mathematics, Budva, Montenegro, 293-302. (2004)
38. Vesin, B., Ivanović, M., Klašnja-Milićević, A., Budimac, Z.: Rule-based Reasoning for Altering Pattern Navigation in Programming Tutoring System. In the proceedings of the 15th international conference on System theory, control and computing, October 14-16, Sinaia, Romania, 644-649. (2011)
39. Vesin B., Ivanović M., Klašnja-Milićević A., Budimac Z.: Rule-based Reasoning for Building Learner Model in Programming Tutoring System, *H. Leung et al. (Eds.): ICWL 2011, LNCS 7048,* Springer, Heidelberg, 154--163. (2011)
40. Vesin, B., Ivanović, M., Budimac, Z.: Learning Management System for Programming in Java. Annales Universitatis Scientiarum De Rolando Eötvös Nominatae, Sectio Computatorica, vol. 31, 75-92. (2009)
41. Vesin, B., Ivanović, M., Budimac, Z., Pribela, I.: MILE - Multifunctional Integrated Learning Environment. In the proceedings of the IADIS Multi Conference on Computer Science and Information Systems MCCSIS'2008, Amsterdam, Netherlands, 104-108. (2008)
42. Vesin B., Ivanović M., Klašnja-Milićević A.,Budimac Z., Protus 2.0: Ontology-Based Semantic Recommendation in Programming Tutoring System*, Experts systems with application, 2012,* DOI: 10.1016/j.eswa.2012.04.052

MSc Boban Vesin has graduated at the Faculty of Science, University of Novi Sad, in 2002. He got his master degree at the same Faculty in 2007 and is working on his PhD thesis. Currently he is a lecturer at Higher School of Professional Business Studies, University of Novi Sad, Serbia. His major research interests are e-Learning and personalization in intelligent tutoring systems. He has published a number of scientific papers in the area.

MSc Aleksandra Klašnja-Milićević is a Lecturer of Computer Science in Higher School of Professional Business Studies at University of Novi Sad, Serbia. She received her Diploma in Electrical and Computer Engineering from Faculty of Technical Sciences at the University of Novi Sad in 2002. She joined the graduate program in Computer Sciences at Faculty of Science, Department of Mathematics and Informatics, University of Novi Sad in 2003,

where she received her M.Sc. degree (2007). Her research interests include recommender systems, information retrieval, user modeling and personalization, and electronic commerce. She has published more than 25 referred papers on her work in national and international conferences and journals.

PhD Mirjana Ivanović since 2002 holds position of full professor at Faculty of Sciences, University of Novi Sad, Serbia. She is head of Chair of Computer Science and member of University Council for informatics. Author or co-author is, of 13 textbooks and of more than 235 research papers on multi-agent systems, e-learning and web-based learning, software engineering education, intelligent techniques (CBR, data and web mining), most of which are published in international journals and international conferences. She is/was a member of Program Committees of more than 120 international Conferences and is Editor-in-Chief of Computer Science and Information Systems Journal.

PhD Zoran Budimac since 2004 holds position of full professor at Faculty of Sciences, University of Novi Sad, Serbia. Currently, he is head of Computing laboratory. His fields of research interests involve: Educational Technologies, Agents and WFMS, Case-Based Reasoning, Programming Languages. He was principal investigator of more than 20 projects. He is author of 13 textbooks and more than 225 research papers most of which are published in international journals and international conferences. He is/was a member of Program Committees of more than 100 international Conferences and is member of Editorial Board of Computer Science and Information Systems Journal.

Received: December 31, 2011; Accepted: July 25, 2012.

DOI:10.2298/CSIS110725002O

A Viewpoint of Tanzania E-Commerce and Implementation Barriers

George S. Oreku[1], Fredrick J. Mtenzi[2], and
Al-Dahoud Ali[3]

[1] Tanzania Industrial Reseach and Deverоment Organizational, P.O.Box 23235,
Kimweri avenue, Dar es salam, Tanzania
gsoreku@tirdo.org
[1] Faculty of Economic Sciences and Information Technology, P.O.Box 1174,
Vanderbijlpark 1900 South Africa.
George.oreku@gmail.com
[2] Dublin Institute of Technology, School of Computing,
Kevin Street, Dublin 8,Ireland
Fredrick.mtenzi@dit.ie
[3] Al- Zaytoonah University, P.O.Box 130,
11733 Amman, Jordan
aldahoud@alzaytoonah.edu.jo

Abstract. The growing rate of ICT utilization particularly the Internet and mobile phones has influenced at an exponential rate online interaction and communication among the generality of the populace. However, with the enormity of businesses on the Internet, Tanzania is yet to harness the opportunities for optimal financial gains. This study is exploratory in nature as it attempts to unveil the prospects of e-commerce implementation, participation, motivation and opportunity to the developing countries like Tanzania where by the domestic market is very big to ensure the growth of agricultural sector. The paper proposes to investigate the ability of consumers to purchase online, the available motivation to do so, and the opportunities for Internet access. We argue the Government and central bank to encourage innovative new technological developments by pre-regulating electronic money to familiarize itself with electronic money schemes generally. Findings revealed that Tanzanians have the ability to participate in e-commerce, but there is need for improved national image to bring in the element of trust and discipline within, and before the international communities. Currently, consumers source for information online but make purchases the traditional way.

Keywords: e-commerce, e-payment, ICT, web, internet access.

1. E-Commerce Phenomenon and Country profile

Tanzania has an area of 945,000 sq km (365,000 sq miles) and a population of about 42 million. Dar-es-salaam is the commercial capital and home to many government institutions and diplomatic missions. There are about 120 ethnic groups on the mainland, although none exceeds 10% of the population, as well as minority Asian and expatriate communities. Tanzania's economy relies heavily on agriculture, which accounts for nearly half of GDP and employs 80% of the workforce. Tourism is growing in importance and ranks as the second highest foreign exchange earner. Mineral production has grown significantly in the last decade and provides over 3% of GDP and accounts for half of Tanzania's exports [14].

The study has shown in [14] by Materu and Diyamett that the use of ICT equipment is still low in Tanzania compared to other countries in the world but it is growing at a staggering pace. According to the World Bank data in the last decade for instance, the penetration rate of personal computers has increased by a factor of 10, while the number of mobile phone subscribers by a factor of 100! Extrapolations until the year 2009 suggests that the penetration rates of personal computers lies around 19.5 computers per 1000 people, which corresponds to an installed base of 850’000 units in 2009.

The Audiencescapes survey of Tanzania was carried out in July 2010 by Tanzania Communications Regulatory Authority (TCRA) as a nationally representative sample. This will allow the researchers to provide accurate breakdowns of urban vs rural use [15]. The survey depicted that Internet use has clearly grown in Tanzania at the rate of 4% but not at the same rate as in neighboring countries like Kenya where the latest estimate of Internet users for Kenya from the ITU is 3995500 people, corresponding to a penetration rate of 9.7%. The table below shows percentage household access amongst those surveyed:

Table 1. Summary of percentage of ICT usage in Tanzania [SourceTCRA]

Media	All sample	Urban	Rural
Radio	85%	85%	84%
TV	27%	59%	14%
Computer	3%	8%	1%
Internet	4%	8%	2%
Mobile phone	62%	82%	54%

It should be noted that although a large number of computers in many cases are owned by the government than private sector but unexpectedly this was vice versa with Tanzania. Many private sectors owned computer compared to government.

The results of TCRA survey have further shown that the average distribution sales of new computers are 50% to government; 40% to the private companies and 10% to private households & small businesses while

the survey from second-hand dealers showed that second hand IT equipment are mainly sold to private households & small businesses. The average life of new computers was found to be 4 years in government and private sector and 8 years in private households and small businesses while the average life of second hand computers was found to be around 5 years.

Based on the results of this survey and some key development statistics for Tanzania, it was estimated that about 200,000 computer units reached their end-of-life in 2009. Future computer mass flow trends as one of the E-Commerce tool based on linear and exponential growth indicate that the potential of E-Commerce implementation is still hindered with many factors as it is depicted in details on this paper.

The number of Internet users around Tanzania has been steadily growing and this growth has provided the impetus and the opportunities for global and regional E-Commerce. However with Internet, different characteristics of the local environment, both infrastructural and socioeconomic, have created a significant level of variation in the acceptance and growth of ecommerce in different regions of Tanzania. It is these controversial finding in the literature that have motivated the paper. The aim of the work is to examine the existing and prospective barriers to E-Commerce to the successful operation of E-Commerce to Tanzanian firms and suggest some strategies to overcome these barriers.

Despite the spectacular dot-com bust of a few years ago, the Internet has markedly changed the way we do business, whether it's finding new streams of revenue, acquiring new customers, or managing a business's supply chain. E-commerce is mainstream — enabling businesses to sell products and services to consumers on a global basis. As such, e-commerce is the platform upon which new methods to sell and to distribute innovative products and services electronically are tested.

The Web's influence on the world's economy is truly astonishing. The business world knows that the Web is one of the best ways for business such as manufacturers to sell their products directly to the public, brick-and-mortar retailers to expand their stores into unlimited geographical locations, and for entrepreneurs to establish a new business inexpensively.

Thus, it is important that the executive in the 21st Century know 1) where technology stands in the business processes of his or her company, 2) how technology relates to the company's strategies, 3) how rapidly technology changes and evolves, and 4) how the company and its business partners will respond to the changing technology.

In the high flying 1990s, many people jumped on the e-commerce bandwagon after reading the many highly publicized dot-com "success" stories. Admittedly, most were written to raise the entrepreneurial blood pressure. What many forgot, though, was the old adage: If it looks too good to be true, it probably is. They didn't use their innate intelligence and failed to proceed with caution.

Nonetheless, the ascendancy of e-commerce has expanded the business environment so that even a small start-up can compete with well-established business names and product brands. Yet, when you consider joining the e-

commerce commerce community, keep in mind that selling products and services on the Web presents a unique set of challenges. This paper will help in identifying and realizing on those challenges with respect to Tanzania scenarios.

There are challenges on what already in place, including a national payment system, local credit cards, and a legislative framework appropriate for e-business. These are challenges that need to be addressed urgently. Most significantly, the legal framework does not provide adequate safeguards to create an environment of trust for e-business transactions to take place. Consequently, financial institutions are not able to set up provisions for supporting e-transactions for their own, and each other's clients. However the use of traditional marketing mechanism is also one of the constraints facing Tanzania participate in e-commerce.

The evidence from literatures also supports that the hype and promise of e-commerce has been well recognized, but the fact is, it has not been realized at the rate which policy documents and government claim. There are very limited ICT developments in Tanzania with less than three people in every 100 people having access to ICT infrastructure [1]. All Governments particularly in Developing countries should play the leading role in the development of Infrastructure including financing Experience has shown that the Private sector is not able to take the responsibility of owning and, thus carrying out all the rehabilitation, and maintenance of the existing network and expansion of the new one that reaches all people in the rural and under-served areas for creating open access to all [21, 22].

There is a very good expectation on e-commerce applications in Tanzania. However there are still some complexities in several aspects such as peripherals like computer importation and use of face to face in conducting business, which brings difficulties in facilitating the take off. The Government should design the framework and policies which may make computer available easy to interested parties within the country, discourage the use of papers in many division from its offices. However it should champion and give the lead to the process. So far it's being claimed that the importation of Computer is tax free but in practice there are difficulties to importing computers to the Country, clearance tariffs are so higher. The situation has not been improved as it is being addressed.

The remainder of this paper is organized as follows: In section 2 the paper describes the literature review. Next we present major barriers of e-commerce development in Tanzania, customer's perceptions in section 3. In section 4 the paper presents the study methodology and Regression model is discussed in section 5. In section 6 the paper presents our recommendations for solutions to the problems and in section 7 we conclude the paper.

2. Literature Review

It is conceived that e-commerce is a phenomenon of developed country and new technology generally put challenges for developing countries that lack the requisite capabilities, as well as the economic and financial resources to cope with the developed countries. Especially internet presents both opportunities for economic and social development, and a threat to further increasing the gap between developed and developing countries [2].

The experience of most developed countries shows that price and availability of the telecommunications infrastructure are clearly associated with competition and market access [3]. Tanzanian Government has withdrawn import duties from computers and computer related peripherals. Due to the withdrawal of duties prices of computers and related products have become affordable to general communities. This to some extent has increased the use of computer for general purpose though effective applications of computers are still underutilized due to particularly government policy. However, it is revealed from recent survey that nearly 90% of the computers are Dar-es-salaam based and there is little scope for decentralization of these PCs to different regions of Tanzania [4].

Very few standard IT institutions are providing high quality IT education in Tanzania, but the costs are very high and consequently remain beyond the reach of ordinary people. Some IT related private institutions opened and started to offer IT courses but again they are centered around big cities such as Dar-es-salaam, Mwanza and Arusha. These institutions suffer from lack of coordination and quality course materials, and inadequate technical facilities. In course of time, eventually a situation has been improved as the government withdrew duties on computers. At present there are more than 50 ISPs operating in the country including the government initiatives of putting in place the fiber optical connecting the whole country [4].

Different patterns have been found in studies about the extent to which firms in developing countries embrace the internet. In Brazil, telecommunication infrastructure is not considered as barrier for e-commerce, and financial services sectors have widely adopted the internet approach [5]. In Nigeria, e-mail was the prime aspect of the internet system and business people used email mostly for the purpose of communication [6]. Low level of IT education was recognized as the underutilization of internet system in many developing countries.

In Hongkong low e-shopping compatibility, e-shopping inconvenience, e-transaction insecurity, and low internet privacy, together with orientation toward social interaction and poor awareness on the part of the consumers, translate into supply-side hurdles [7].

It is found from various studies that in developing countries e-commerce has hindrances in the arena of cultural habit and business and technology infrastructures as well [8].

Various studies identified a number of factors that facilitate or limit internet-based businesses. The enablers are availability of information, access to price information, accessibility, and convenience. These are the

factors that would benefit the online business. On the other hand, the limiters which inhibit the escalation of internet business include lack of trial, lack of interpersonal trust, lack of instant gratification, high shipping and handling costs, customer service issues, loss of privacy and security, lack of a stable customer base, and poor logistics. Oinas in [9] recommended in his paper that online companies serving ultimate consumers need to build competency in retailing, handling payments, and distribution, among other crucial business functions [9].

3. Major Barriers of e-Commerce Development in Tanzania: Customers Perceptions

E-commerce is ubiquitous and thus anyone can transact at any time from any place. On-line commerce has enabled customers to overcome the handicaps of time and space. However, despite the rapid and demonstrated uptake of e-commerce techniques, there is still very limited detailed evidence about how individual corporations in developing countries are using e-commerce to improve their business activities and what the effective costs and benefits are of using those techniques (*Digital Opportunities for Development*). Despite the fact that e-commerce has endless opportunities, it is evident that numerous barriers inhibit the successful uptake of e-commerce as can be referred from figure one above. One point of this paper is to revealing the existing and prospective barriers to e-commerce and devising their solutions in the context of Tanzania.

3.1. Context: Tanzania

Consequently, there appears to be major problems in defining 'E-commerce', generally in the entire Tanzanian context. But according to the WTO, “Electronic commerce refers to the production, distribution, marketing, sale or delivery of goods and presentation of electronic services.” Thus, electronic commerce or e-commerce is understood as all commercial activities on electronic networks, including promotion, online sale of products and services, customer service, etc.

According to International Telecommunication Union (ITU) report, there are 520,000 Internet users in Tanzania as of June, 2009, 1.3% of the population, according to 2010 ITU report. There are around hundreds of formal and informal IT training centers and numerous computer shops. Although ICT had been announced as a thrust sector in 2003 year no substantial and clear-cut IT policy has been followed since then. Still legislation towards electronic signatures, practical laws to protect intellectual property rights and relevant financial structure to facilitate electronic transaction are yet to be formalized. The entry into the global economy is effectively blocked because of inadequate ICT infrastructure and human

resources, and non-existing compatible electronic environment to the rest of the world, lack of coordination among different stakeholders. However, the member of IT users in Tanzania is increasing rapidly.

3.2. Technical Limitations to e-Commerce

There is no doubt behind the fact that E-commerce has given many companies the right to cheer but there are few limitations of E-commerce too. Hence understanding the drivers and barriers of e-commerce adoption becomes increasingly important as can be observed from different authors [18].

Few technologies have realized the many benefits e-commerce does, whether taking a small business to never before seen global proportions or opening up millions of new customer markets. E-Commerce has given many companies the right to cheer, but Tanzania these have not taken place and here are a few of the reason why:

- Lack of sufficient system security, reliability, standards and communication protocols,
- Insufficient telecommunication bandwidth,
- The software development tools are still evolving and changing rapidly,
- Difficulties in integrating the internet and e-commerce software with some existing applications and databases,
- The need for special web servers and other infrastructures, in addition to the network servers (additional cost),
- Possible problems of inter operability, meaning that some E-commerce software does not fit with some hardware, or is incompatible with some operating systems or other components.

3.3. Non Technical Limitations to e-Commerce

Despite the fact of the mentioned technical limitations above, E-Commerce also has its own limitations in non-technological as follow:

- Cost and Justification,
- Security and privacy,
- Lack of trust and user resistance,
- Channel conflict,
- Other limitations factors are such as lack of touch and feel online etc.

According to the study conducted by Oreku et al., in [10] E-Commerce readiness in Tanzania is not advancing because of;

- Poor physical and network infrastructures,
- Inadequate human resources,

- Absence of required rules,
- Low level of computer literacy,
- Widespread poverty etc.

4. Study Methodology

The study methodology followed to complete the study is on the basis of primary and secondary data. The result from this study was collected from different sources. Secondary data were collected from relevant papers, daily newspaper, and IT magazines published in paper form and electronic form as well. Primary data were collected from interviews from five major Bank Managers "(Tanzania Postal Bank ,Cooperative Rural Development Bank ,National Bank of Commerce ,National Microfinance Bank ,Azania Bank);" 14 stakeholder groups namely, vendors (merchants) with not less than thirty employees, 3 financial institutions, Top five IT Mangers to the institutions and the group of seventy nine consumers (mostly SMEs). A critical analysis was done to determine the barriers that hinder the effective implementation of e-commerce in Tanzania.

Among other barriers many traditional middlemen are trying to preserve existing barriers and create new ones as a way to prevent online competition. In the developed countries these barriers already prevented many firms practicing e-commerce from selling directly to consumers and severely limit the ability of consumers to buy things.

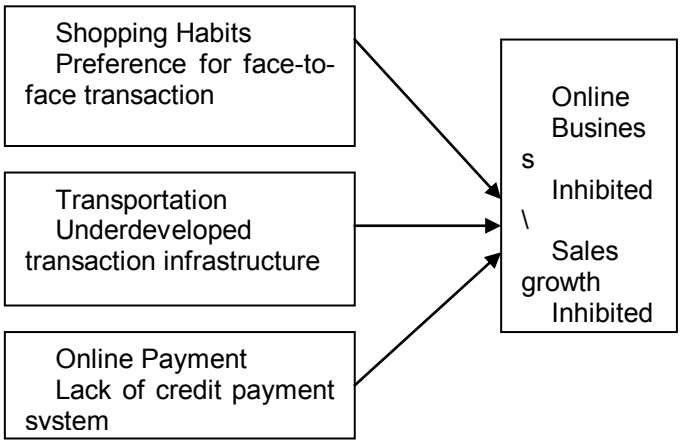


Fig. 1. Barriers to Online Business [source Author]

The challenge confronting Tanzania is to create an ideal market structure for E-Commerce that will stimulate and modernize network development and infrastructure such as currier services; accelerate universal access; support

affordable access; encourage investment and innovation which will mean more business. Because of the critical nature of these issues, government and the business community are faced with the challenges of developing strategies and policies that will strengthen the infrastructure needed to support effective use of e-commerce.

4.1. Study Findings

The study collected and analyzed primary data about existing and prospective inhibitors from customers. The study has investigated six critical factors namely: lack of security, lack of privacy, lack of information, lack of experts, computer illiterates and inappropriate law.

From study undertaken six critical factor identified from questionnaires were able to indicate the sense of barriers to the growth of E-Commerce in Tanzania. Since the number of observations for the study was relatively small, the results provide some general ideas about the directions of the three hypothesized relationships:

I. The current patterns of E-commerce activities may change as E-commerce matures, and Internet infrastructures in many countries are improved (Lynch and Beck, 2001).

III. When products and services are available online and in high demand by other countries, there is a possibility that foreign sales for the products and services would be increased.

III. Internet accessibility in the home country will negatively moderate the relationship between Internet and E-commerce capabilities and the proportion of export sales to total sales of Firms.

Table 2. Model Summary of six E-Commerce implementations factors.

		Lac_of _sec	Lac_of _pri	Lac_of _inf	Lac_of _exp	Comp _ill	Inap _law
N	Valid	200	200	200	200	200	200
Mean		2.9900	2.4500	2.7000	2.8500	2.3500	2.1500
Median		3.0000	2.0000	3.0000	3.0000	2.0000	2.0000
Std.dev		1.1904	1.1904	1.1298	1.1723	.9550	.8551

The mean score of the variable lack of security which is the observations from lowest value to highest value and picking the middle shows that the average people to some extent agree about the fact that it has substantial contribution to the obstacles of E-Commerce. The mean score of the lack of experts, computer illiteracy, and inappropriate laws indicates that the average respondents agreed that these variables have impact on the development of e-commerce in Tanzania.

5. Regression Model

Electronic commerce (e-commerce) is a growing field of scholarly research especially in information systems, economics and marketing, but it has received little to no attention in statistics. This is surprising because it arrives with an enormous amount of data and data-related questions and problems[19,20] In light of the special data structures collected from the field survey we analyse these functional data which can play major role in this field through regression models.

In this study, the dependent variable "inefficient e-Commerce" which indicates ineffi_e_Commerce and independent variables were: (a) inappropriate laws indicates as inap_law (b) computer illiteracy indicates as comp_ill, (c) lack of experts indicates as lac_of_exp, (d) lack of infrastructure indicates as lac_of_inf, (e) lack of privacy indicates as lac_of_pri, (f) lack of security indicates as lac_of_sec.

The model summary contains six models. Model 1 refers to the first stage in the hierarchy when only inappropriate law is used as a predictor. Model 2 refers to the second stage in the hierarchy when inappropriate law and computer illiteracy are used as predictors. Model 3 refers to the third stage in the hierarchy when inappropriate law computer illiteracy and lack of expert are used as predictors. Model 4 refers to the fourth stage in the hierarchy when inappropriate law, computer illiteracy, lack of expert, and lack of infrastructure are used as predictors and so on.

In the column labeled R are the values of the multiple correlation coefficients between the predictors and the outcome. When only inappropriate laws is used as predictor, this is the simple correlation between inefficient e-commerce system and inappropriate laws (0.294), when inappropriate laws and computer illiteracy are used as predictors the simple correlation between inappropriate laws and computer illiteracy (0.301) and so on for other predictors.

The next column gives a value of R^2 which is a measure of how much of the variability in the outcome is accounted for by the predictors. For the first model its value is 0.087, which means that inappropriate law as predictor accounts for 8.7 per cent of the variation in the dependent variable inefficient e-commerce. The values of second, third, fourth, fifth, and sixth models increase to 9.1%, 12.4%, 13%, 13.5%, and 16%. The adjusted R^2 gives some idea of how well model generalizes and ideally it would like its values to be the same or very close to the value of R^2. The difference for the final model is a fair bit (0.160-0.134=0.026 or 2.6%). This means that if the model was derived from the population rather than a sample it would account for approximately 2.6% less variance in the outcome. The Durbin-Watson statistic informs about whether the assumption of independent errors is tenable. The closer to that the value is, the better, and for these data the value is 2.011, which is so close to 2 that the assumption has almost certainly been met.

Table 3. Regression Model Summary.

					Change Statistics					Durbin Watson
Model	R	R^2	Adjusted R square	St. Error of the Estimate	R. Square Change	F. Change	df 1	df 2	Sig. F. Change	
1	0.294[a]	0.087	0.082	0.44018	0.087	18.764	1	198	0	
2	0.301[b]	0.091	0.081	0.44033	0.004	0.864	1	197	0.354	
3	0.353[c]	0.124	0.111	0.43315	0.034	7.583	1	196	0.006	
4	0.36[d]	0.13	0.112	0.43296	0.005	1.174	1	195	0.28	
5	0.368[e]	0.135	0.113	0.43263	0.006	1.299	1	194	0.256	
6	0.4[f]	0.16	0.134	0.42749	0.025	5.691	1	193	0.018	2.011
a) Predictors: (Constant), inap_law										
b) Predictors: (Constant), inap_law, comp_ill										
c) Predictors: (Constant), inap_law, comp_ill, lac_of_exp										
d) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf										
e) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf, lac_of_pri										
f) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf, lac_of_pri, lac_of_sec										
g) Dependent variable: ineffi_e_Commerce										

Where:

R - Values of multiple correlation coefficients, R^2 – Outcome variability, F – Frequency, df- Difference, Sig-Significant

The next part of the output contains an analysis of variance (ANOVA) that test whether the model is significantly better at predicting the outcome than using the mean as a 'best guess'. Specifically, the *F*-ratio represents the ratio of the improvement in prediction that results from fitting the model (labeled 'Regression in the table'), relative to the inaccuracy that still exists in the model ('Residual' in the table). This table is again split into six sections: one for each model.

The regression model is much greater than the inaccuracy within the model then the value of *F* will be greater than 1 and SPSS calculates the exact probability of obtaining the value of *F* by chance. For the initial model the *F*– ratio is 18.764, which is very unlikely to have happened by chance ($p<.001$). For the second model the value of F is 9.808, which is also highly significant ($p<.001$). The value of *F*- ratio of third, fourth, and sixth models are 9.285, 7.263, 6.079, and 6.137, which are also highly significant ($p<.001$). we can interpret these results as meaning that the final model may count as significant to predict the outcome variable.

Table 4. Analysis of variance (ANOVA).

Model		Sum of Square	df	Mean Square	F	Sig.
1	Regression	3.636	1	3.636	18.768	.000[a]
	Residual	38.364	198	.194		
	Total	42.000	199			
2	Regression	3.803	2	1.902	9.808	.000[b]
	Residual	38.197	197	.194		
	Total	42.000	199			
3	Regression	5.226	3	1.742	9.285	.000[c]
	Residual	36.774	196	.188		
	Total	42.000	199			
4	Regression	5.446	4	1.362	7.263	.000[d]
	Residual	36.554	195	.187		
	Total	42.000	199			
5	Regression	5.689	5	1.138	6.079	.000[e]
	Residual	36.311	194	.187		
	Total	42.000	199			
6	Regression	6.729	6	1.122	6.137	.000[f]
	Residual	35.271	193	.183		
	Total	42.000	199			
a) Predictors: (Constant), inap_law						
b) Predictors: (Constant), inap_law, comp_ill						
c) Predictors: (Constant), inap_law, comp_ill, lac_of_exp						
d) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf						
e) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf, lac_of_pri						
f) Predictors: (Constant), inap_law, comp_ill, lac_of_exp, lac_of_inf, lac_of_pri, lac_of_sec						
g) Dependent variable: inefficient e-Commerce						

The next part of the output is concerned with the parameters of the model. The first step in the hierarchy included inappropriate laws and although these parameters are interesting up to a point, it is more interested in the final model because this includes all predictors that make a significant contribution to predicting relationship between predictors and inefficient E-Commerce in Tanzania. It will actually look only at the lower half of the table (Model 6).

In multiple regressions the model takes the form of an equation that contains a coefficient (b) for each predictor. The first part of the table gives us estimates for these b values and these values indicate the individual contribution of each predictor to the model.

The B values tell us about the relationship between inefficiency and each predictor. If the value is positive it can tell that there is a positive relationship between the predictor and the outcome whereas a negative coefficient represents a negative relationship. For these data predictors have positive b values indicating positive relationships. So we see that the more inappropriate law the more inefficient will be the state of E-Commerce and affect outcome if the effects of all other predictors are held constant.

Table 5. Coefficients

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	1.360	.084		16.114	.000
	inap_law	.158	.036	.294	4.332	.000
2	(Constant)	1.395	.092		15.119	.000
	inap_law	.184	.046	.342	4.022	.000
	comp_ill	-.038	.041	-.079	-.930	.354
3	(Constant)	1.245	.106		11.768	.000
	inap_law	.183	.045	.340	4.075	.000
	comp_ill	-.065	.041	-.136	-1.578	.116
	lac_of_exp	.076	.027	.193	2.754	.006
4	(Constant)	1.165	.129		9.050	.000
	inap_law	.181	.045	.337	4.027	.000
	comp_ill	-.065	.041	-.135	-1.566	.119
	lac_of_exp	.077	.027	.196	2.794	.006
	lac_of_inf	.029	.027	.073	1.084	.280
5	(Constant)	1.084	.147		7.355	.000
	inap_law	.179	.045	.333	3.985	.000
	comp_ill	-.060	.042	-.125	-1.449	.149
	lac_of_exp	.078	.027	.200	2.849	.005
	lac_of_inf	.024	.028	.059	.867	.387
	lac_of_pri	.035	.031	.078	1.140	.256
6	(Constant)	.988	.151		6.539	.000
	inap_law	.166	.045	.308	3.705	.000
	comp_ill	-.050	.041	-.105	-1.224	.222
	lac_of_exp	.059	.028	.151	2.100	.037
	lac_of_inf	.024	.027	.060	.893	.373
	lac_of_pri	.018	.031	.041	.593	.554
	lac_of_sec	.065	.027	.169	2.386	.018

Each of these B values has an associated standard error indicating to what extent these values would vary across different samples, and these standard errors are used to determine whether or not the *B* value differs significant from zero. Therefore, if the t-test associated with a *B* value is significant (if the value in the column labeled sig. is less than 0.05) then that predictor is making a significant contribution to the model. For this model 6, inappropriate law is equal to (3.705), *sig*<.01, lack of experts is taken as equal to (2.100), sig<.05 and lack of security is equal to (2.386), *sig*<.05 which are all significant predictors of inefficient E-Commerce. From the magnitude of the t-statistics we can see that the inappropriate law had more impact than lack of experts and lack of security.

The standardized beta values (β) are all measured in standard deviation units and so are directly comparable: therefore, they provide a better insight into the 'importance of predictor in the model the standardized beta values for inappropriate laws (.308), computer illiteracy (-.105), lack of experts (.151), lack of infrastructure (.060), lack of privacy (.041) and lack of security (.169).

It reveals except for computer illiteracy all other variables are positive. Therefore, interestingly the computer illiteracy is not a good predictor of inefficiency of e-Commerce in Tanzania.

6. Recommendation for Solution to the Problems

Since Tanzania is developing country and private organizations are not organized enough to provide with IT infrastructure government should initiate programs to reduce the barriers. Some of recommended initiatives could be establishing a task force at the government level to coordinate the ICT related activities to different stakeholders. As a long-term investment government should invest in basic and higher education to reap the real benefits of ICT [13].

The E-Commerce innovation programme will build capacities in Tanzania to small and medium ICT enterprises to make a business with ICT utilizations. E-commerce innovation aims to encourage the growth of Tanzanian ICT industries and SMEs, particularly in selected regions, Dar-es salaam, Arusha, Mwanza, Morogoro and southern regions through three main actions: Strengthening and improving security models for e-commerce in Tanzania in banking systems, fostering SMEs groups use of ICT and supporting innovative local applications i.e. websites sustainability and Single government institution managed portals development.

An effective telecommunications infrastructure to facilitate export oriented IT services is to be taken as a must at the moment. Government should subsidize utility expenses for IT companies and declare tax holiday for IT and IT education enterprises. Level of English education is to be upgraded to the communication skills of the human resources. Tanzanian skilled diasporas can be encouraged to return to the country and/or collaborate with Tanzanian entrepreneurs.

The growth of e-commerce depends on broad and affordable access to infrastructure, enabled by convergence of technologies, forward looking telecommunications policy, robust network infrastructure, sufficient bandwidth and support for targeted applications. The infrastructure foreseen for e-commerce in Tanzania, against the background of globalization, should be capable of handling many services and applications. The utilizations of telecommunications infrastructure available and the availability of and access to broadband infrastructures will be important in driving the necessary innovation in e-commerce services.

The lack of e-payment system is one of the main hindrances to e-commerce. Most of the IT activities particularly transactions with other countries require e-payment system badly. For example a single Paypal would be a great aid to solve the payment systems problems but surprisingly enough to date Tanzania is not Paypal's list.

E-Commerce is growing for several reasons. Despite many advantages there is a dilemma for vendors to really capture and rip the benefit of it. In

this part of the paper we have outlines simple steps to take to ensure the success of an E- Commerce business:

- Reducing Consumer Reluctance for Online Shopping,
- Careful selection of products to offer in the virtual stores in terms of nature and price of the products,
- Product standardization,
- Educating consumers about the ease and benefits of online shopping,
- Considering the value that the customers consider while delivering goods about the benefits the consumer gets from possessing and using a product and the associated costs for acquiring the product [11],
- Substantially enhancing transaction security and product quality, showing the customers that the company cares and shares about buyers' well-being is instrumental to enhancing customer loyalty [12] and to help them understand that virtual shops are safe and legitimate,
- Building effective distribution channels namely postal service, direct delivery, third party delivery, and alliances with other established companies,
- Removing any obstacles that hinder the effective methods of both online and offline payment systems,
- It is imperative that the WTO support barrier-free e-commerce and the WTO rules and disciplines are applied, and where necessary adapted, to ensure effective execution of e-commerce. Adopting and implementing the WTO Information Technology agreement on financial services and the WTO agreement on basic Telecommunications are essential for international business relating to e-commerce (Worldwide Coalition Calls for WTO Policy Agenda to Enhance Growth of E-Commerce).

6.1. Suggestions to the Banks and Policy Makers

Although the Tanzania Government and Central Banks has not taken the lead in publishing its position on electronic money, it is proposed that the following issues to be investigated.

- Many countries are currently in the process of adopting "digital cheques" purporting to fulfill the same function as traditional paper based cheques. Tanzanian and central Bank should also find the possibility of applying to and/or regulating to "digital cheques". The Tanzania inability of applying to digital cheques constitutes a material barrier to E-commerce due to the absence of adequate consumer protection and commercial certainty of payment provided. The Banks however should be of the opinion that the use of credit pushes instruments for low value retail payments should be encouraged instead of debit pull instruments. With "credit push" the payer initiates the transfer of funds to the payee whereas the "debit pull" requires the recipient to collect the funds from the payer's bank.
- The Prevention of Counterfeiting of Currency in its current form is incapable of being applied to the "counterfeiting" of electronic money.

- The issuance of electronic money may fall outside the definition of "business of a bank". Accordingly, issuers of electronic money may find themselves "unregulated" and consumers "unprotected". However, only banks would be allowed to issue electronic money from now on. Primary and intermediary issuers of electronic value will therefore be subject to regulation and supervision by the Tanzanian Central Bank. Although single purpose schemes will generally fall outside the definition of electronic money, the Tanzanian Central Bank will determine whether multi-purpose schemes fall within the definition or not.

6.2. Questions for Policy consideration

Section 6.2 of the paper is aimed at policy makers who are involved in the development or management of programmes in the ICT sector in developing countries. It provides a 'snapshot' of the E-Commerce interventions questions that should be considered in the E-Commerce sector development and the policy debates during E-Commerce discussions. It draws from the experience of use in both the North and South of Africa, but with a focus on applicability in Tanzania to identify the most effective and relevant way to reinforce E-Commerce to the country,

1. What steps need to be taken to further upgrade and integrate national financial services infrastructure so as to facilitate E-commerce?
2. How can basic banking services be extended to the broader population, to allow use of electronic payments, credit, and funds transfers?
3. What types of electronic payment systems and technology are most appropriate and practical? How can these be developed effectively on a national level, in co- ordination with international industry efforts?
4. How should the government support these development efforts, both logistically and financially? Which agencies should be responsible? Are these legislative actions that need to be considered?
5. Should non-banking institutions be allowed to issue e-money? How can the Central Bank ensure that such non-banking institutions are licensed, regulated and prudentially secured?

7. Conclusion

Despite a few stumbles, the future is bright for e-commerce. The 20th Century, shaped by the Industrial Revolution, became the age of the automobile and the television. The 21st Century, shaped by the Technological Revolution, is the age of globalization. The Internet massively impacts all aspects of business. In the 21st century, e-business is no longer an option for businesses; it is a necessity.

In the study the authors intended to examine the existing and prospective barriers to e-commerce to the successful operation of e-commerce in

Tanzania and suggest some strategies to overcome these barriers. Companies that market to Tanzania customers on the internet need to devise some unique ways to overcome the constraints that suit indigenous environment.

Today, e-commerce is an ever-expanding consumer industry. For an e-commerce site to succeed it must understand its customers' mindset. Although price is always an issue, it is rarely the primary motivator for buying a product online. Customers are looking for convenience, and/or products they can't find elsewhere. Vendors should not wait until the removal of the current obstacles in the online business environment. The effort is to be exerted towards the development of appropriate e-commerce model that is suitable for the products being marketed. The business model has to encompass the three major factors: attracting potential customers, timely delivery, and comfortable payment methods.

Tanzania is an agricultural country. The country should take the approaches to e-commerce holistically and would exert efforts to the proper utilization of ICT particularly, agricultural e-Commerce.

Small websites that cater to niche markets have the best chance of prospering. That is, as long as you take care to ensure that your customers' shopping experiences aren't marked with too many potholes. The Entrepreneurs can come together to let companies and government know that they won't tolerate the artificial barriers that limit choice and raise prices. They can work with industries, professional associations together and realize the promise of e-commerce and not to blocking it.

References

1. Faria J.A Tanzania should embrace e-Commerce, says UN official, retrieved on 6th December 13, http://www.i4donline.net/news/news-details.asp?newsid=1449 (2010)
2. Sachs, J. D., Readiness for the networked world: A guide for developing countries. Cambridge, MA: Center for International Development, Harvard University (2000).
3. Bjørn F "A Rural-urban digital divide? Regional aspects of Internet use in Tanzania." Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries, São Paulo, Brazil, May (2007).
4. Osuagwu L., "Internet Appreciation in Nigerian Business Organizations", Journal of Internet Commerce, 2 (1), 29-47, (2003).
5. Cheung, Michael T. and Liao Z. Supply-Side Hurdles in Internet B2C E-Commerce, IEEE Transactions on Engineering Management, 50 (4), 458-469, (2003).
6. Jun Y. B2C Barriers and Strategies: A Case Study of Top B2C Companies in China, Journal of Internet Commerce, 5:3, 27 — 51, (2006).
7. Oinus, Paivi, Towards Understanding Network Relationships in Online Retailing, International Review of Retail, Distribution, and Consumer Research, 12 (3), 319-335 (2002).

8. Oreku G.S, Jiazhong L.,Mtenzi F and Kimeli V., State of Tanzania e-readiness and e-commerce: Overview, Information Technology for Development, Volume pages 302–311, autumn (Fall) (2009).
10. Armstrong, Gary and Phillip Kotler Marketing: An Introduction (6th ed.). New Jersey: Prentice Hall,(2002).
11. Srinivasan, Srini S., Rolph Anderson, and Kishore Ponnavolu , "Customer Loyaltyin E-commerce: An Exploration of Its Antecedents and Consequences", Journal of Retailing, 78 (1), 41-50. 2002.
12. Khanam D.,Ahmed M., Husain S. Khan, E-Banking: An Emerging Issue of Developing Country Like Bangladesh", Pakistan Journal of Social Sciences, Grace Publication, 3 (3):, 526-529, (2005)
13. Lynch, P.D. and J.C. Beck, "Profiles of Internet Buyers in 20 Countries: Evidence for Region-Specific Strategies,"*Journal of International Business Studies*, Vol. 32, No. 4: 725-748, 2001.
14. Materu M.B. and Diyamett B.D "Towards Evidence-based ICT Policy and Regulation" Volume Two, Policy Paper 11, (2010)
15. Report on Internet and data services in Tanzania, A supply –Side Survey TCRA September 2010
16. Tigre, P.B., "E-Commerce Readiness and Diffusion: The Case of Brazil", I-WAYS, Digest of Electronic Commerce Policy and Regulation, 26, 173-183, (2003).
17. Tigre, P.B.,, Brazil in the Age of Electronic Commerce, The Information Society, 19:1, 33 — 43, (2003).
18. Zhu, K, Kraemer, K & Xu, S 2003, 'Electronic business adoption by European firms: a cross-country assessment of the facilitators and inhibitors', European Journal of Information System, vol. 12, pp. 251-268.
19. J.Wolfgang and PK Kannan STATISTICAL METHODS IN ECOMMERCE RESEARCH Chapter: Dynamic Spatial Models in Online Markets, Book Chapter A JOHN WILEY & SONS, INC., PUBLICATION, July 2007.
20. J.Wolfgang and S. Galit , Functional Data Analysis in Electronic Commerce Research, Statistical Science, 2006, Vol. 21, No. 2, 155–166).
21 Tanzania Country Paper as Input to the 2nd World Summit on the Information Society Preparatory Committee (WSIS Prepcom ii), Geneva, 2003-Tunis 2005, cited from www.itu.int/wsis/docs/pc1/.../tanzania.doc,June 2012
22 Ndou, V.D.,E – Government For Developing Countries: Opportunities and Challenges, EJISDC (2004) 18, 1, 1-24

George S. Oreku is a Principal researcher and Director of ICT and Technology Transfer with TIRDO. He is also a Post Doctoral researcher with North West University in South Africa. He has worked as a lecturer and as an external examiner in many Universities Worldwide. He has organized and chaired a number of International Workshops and Conferences. He is a reviewer in many International Journals and Conferences as well as a member of IEEE, ACM, SANORD, ERB and WASET.

Fred Mtenzi is a Lecturer at the School of Computing, Dublin Institute of Technology, Ireland. Prior to joining DIT, he worked as a Lecturer at the University of Dar es salaam in Tanzania. His research interest includes design of algorithms for solving combinatorial optimisation problems, energy aware routing in mobile ad hoc networks and its related security issues, cybercrime, pervasive computing and knowledge management. He has organised and chaired a number of international conferences. He has been a Guest Editor in a number of journal special issues. He is a member of the IEEE, ACM, and ISSA.

AL-Dahoud Ali, is a full professor at Al-Zaytoonah University, Amman, Jordan. He took his High Diploma from **FON** University Belgrade 1986, PhD from La Sabianza1/Italy and Kiev Polytechnic/Ukraine, on 1996. He worked at Al-Zaytoonah University since 1996 until now. He worked as visiting professor in many universities in Jordan and Middle East, as supervisor of master and PhD degrees in computer science. He established the ICIT conference since 2003 and he is the program chair of ICIT until now. He was the Vice President of the IT committee in the ministry of youth/Jordan, 2005, 2006. Al-Dahoud was the General Chair of (ICITST-2008), June 23–28, 2008, Dublin, Ireland (www.icitst.org). He has directed and led many projects sponsored by NUFFIC/Netherlands, and Spanish Agency for International Development Cooperation. His hobby is conference organization, so he participates in the following conferences as general chair, International Chair, program chair, session's organizer or in the publicity committee: ICITs, ICITST, ICITNS, DepCos, ICTA, ACITs, IMCL, WSEAS, AICCSA and CCSIE 2011. **Journals Activities**: Al-Dahoud worked as Editor in Chief or guest editor or in the Editorial board of the following Journals: Journal of Digital Information Management, IAJIT, Journal of Computer Science, Int. J. Internet Technology and Secured Transactions, and UBICC. He published many books and journal papers, and participated as keynote speaker in many conferences worldwide.

Received: July 25, 2011; Accepted: July 22, 2012.

DOI:10.2298/CSIS111102003A

A Design Specification and a Server Implementation of the Inverse Referential Integrity Constraints

Slavica Aleksić[1], Sonja Ristić[2], Ivan Luković[1], and Milan Čeliković[1]

[1]University of Novi Sad, Faculty of Technical Sciences,
Department of Computing and Control
Trg Dositeja Obradovića 6
21000 Novi Sad, Serbia
{slavica, ivan, milancel}@uns.ac.rs
[2]University of Novi Sad, Faculty of Technical Sciences,
Department for Industrial Engineering and Management
Trg Dositeja Obradovića 6
21000 Novi Sad, Serbia
[2]sdristic@uns.ac.rs

Abstract. The inverse referential integrity constraints (IRICs) are specialization of non-key-based inclusion dependencies (INDs). Key-based INDs (referential integrity constraints) may be fully enforced by most current relational database management systems (RDBMSs). On the contrary, non-key-based INDs are completely disregarded by actual RDBMSs, obliging the users to manage them via custom procedures and/or triggers. In this paper we present an approach to the automated implementation of IRICs integrated in the SQL Generator tool that we developed as a part of the IIS*Studio development environment. In the paper the algorithms for insertion, modification and deletion control are presented, alongside with parameterized patterns for their implementation for DBMSs MS SQL Server 2008 and Oracle 10g. It is also given an example of generated procedures/triggers.

Keywords: Inclusion Dependencies, Inverse Referential Integrity Constraint, Declarative Constraint Specification.

1. Introduction

A common approach to database design is to describe the structure and constraints of the Universe of Discourse (UoD) in a semantically rich conceptual data model. The Entity-Relationship (ER) diagrams or the UML (Unified Modelling Language) class diagrams are widely used to represent the conceptual database schemas. The obtained conceptual database (DB) schema is translated latter on into a logical DB schema, representing a design specification of the future database. Such design specification is to be

implemented by means of a database management system (DBMS). Contemporary DBMSs are mostly based on the relational or object-relational data models. Therefore, logical DB schemas are still expressed by the concepts of relational data model. Furthermore, logical DB schemas as the design specifications are normally transformed into error free SQL specifications of relational or object-relational DB schemas. In this way, a designed database may be implemented. These SQL specifications are implementations of the structure and constraints of UoD specified in the conceptual DB schema. A goal of this paper is to present an approach to the specification and implementation of a relational integrity constraint type called the **inverse referential integrity constraint** (IRIC).

The most fundamental integrity constraints that arise in practice in relational databases are functional dependencies (FDs) and inclusion dependencies (INDs). There are two basic kinds of INDs: key-based INDs and non-key-based INDs. More often key-based INDs are called referential integrity constraints (RICs). On the contrary, IRICs are a kind of non-key-based INDs. More details about INDs, as well as definitions of different kinds of INDs, including the IRICs, are given in Section 3.

In ER data model or UML class meta-model, cardinality or multiplicity constraints are used, among all, to express the existential dependency between two entity types, i.e. classes. Namely, the existential dependency is modelled by setting the minimal multiplicity to one. Such existential dependency between two entity types in a conceptual DB schema causes an IRIC to be specified in a relational DB schema, as its consequence. More precisely, an IRIC specification in a relational DB schema is caused by a minimal multiplicity set to one, together with the maximal multiplicity set to many on the same side of the association between the two entity types in a conceptual DB schema.

While the referential integrity constraints may be fully enforced by most current relational database management systems (RDBMSs), non-key-based INDs are completely disregarded by actual RDBMSs, obliging the users to manage them via stored program units and triggers. This implies an excessive effort to maintain integrity and develop applications.

There are numerous contemporary software tools aimed at an automated conceptual database schema design and its implementation under different (mostly relational or object-relational) database management systems, such as: DeKlarit, ERwin Data Modeler, Oracle Designer, Power Designer etc. Some of them are described in [7], [14], [24], [28]. All of them enable setting the relationship minimal multiplicity to one. Therefore, they support the specification of the existential dependency between two entity types in the conceptual database schema. However, all of them ignore this specification when generate the SQL code to implement a relational or an object-relational database schema. Even more, to the best of our knowledge, neither of the other CASE tools offers such functionality, as well. As a rule, they do not employ any procedural DBMS mechanisms to provide the automatic implementation of IRICs.

Our approach to the specification and implementation of the IRICs is implemented through the development environment IIS*Studio (IIS*Studio DE, current version 7.1). The development of IIS*Studio DE is spanned through a number of research projects lasting for several years, in which the authors of the paper are actively involved. One of its integral parts is Integrated Information Systems*Case (IIS*Case) – a software tool that supports a model driven approach to information system (IS) design. It supports conceptual modelling of database schemas and generating executable application prototypes. A case study illustrating main features of IIS*Case is given in [19]. Methodological aspects of its usage may be found in [20]. A description of information system design and prototyping using form types is given in [25].

Many commercial CASE tools, e.g. ERwin Data Modeler, Oracle Designer, Power Designer, use ER data model or UML class meta-models to express a conceptual schema. Unlike them, IIS*Case provides a specific platform independent meta-model that does not rely on the ER or UML meta-models. Among the other, this meta-model provides the concepts of form types, component types and their attributes, at the abstraction level of a conceptual DB schema.

The attribute and the form type concepts are explained in details in [19] and [26]. The multiplicity constraints are included in the set of constraints that may be specified by means of form types. IIS*Case uses the set of attributes and the set of form type specifications as the input data for database design to generate logical DB schemas as 3rd normal form (3NF) relational DB schemas with all the relation scheme keys, null value constrains, unique constrains, referential and inverse referential integrity constraints, derived from an IIS*Case conceptual data model. These schemas are stored in the IIS*Case repository. The specification of the IIS*Case repository is given in [25].

In order to provide an efficient transformation of design specifications into error free SQL specifications of relational database schemas we developed the SQL Generator [2]. It is a tool that utilizes SQL, as one of the most common domain-specific languages applied at the level of DB servers. One of the main reasons for the development of such a tool was to make DB designer's and developer's job easier, and particularly to free them from manual coding and testing of SQL scripts for the creation of tables, views, indexes, sequences, procedures, functions and triggers. The SQL Generator implements one transformation in the chain of all IIS*Case transformations from the conceptual model, which is platform independent, towards the executable program code. The input into SQL Generator is a relational database schema, obtained by a transformation of the conceptual DB schema and stored in the repository.

Our SQL Generator implements constraints of the following types: domain constraints, key constraints, unique constraints, tuple constraints, native and extended referential integrity constraints, referential integrity constraints inferred from nontrivial inclusion dependencies, and inverse referential integrity constraints ([18], [23]). Constraints are implemented by the

declarative DBMS mechanisms, whenever it is possible. However, the expressiveness of declarative mechanisms of commercial DBMSs may be limited. Therefore, SQL Generator implements a number of constraints through the procedural mechanisms [3]. In this paper we present a feature of SQL Generator that provides an automated implementation of IRICs that are caused by the multiplicity specifications in the IIS*Case conceptual model.

Apart from the Introduction and Conclusion the paper has five sections. Section 2 presents the related work. In Section 3 the notion of an IRIC is explained, illustrated with a real life example to point out the necessity of IRICs implementation. The algorithms for insertion, modification and deletion control in the presence of IRICs are presented in Section 4. In Section 5 we present parameterized patterns of the aforementioned algorithms for DBMSs MS SQL Server 2008 [21] and Oracle 10g [24]. In [4] we introduce patterns for the insertion of mutually blocked tuples via a view created over the relations $r(N_i)$ and $r(N_j)$. Apart from these patterns, here in Section 5, we also present in details patterns for the insertion of mutually blocked tuples via custom db procedures. In Section 6 we present an example of an IRIC design specifications and transformation of design specifications into error free SQL specifications of relational DB schemas by means of IIS*Studio.

2. Related work

Integrity has always been an important issue for database design and implementation. Its importance grows with increasing demands according the quality and reliability of data. Integrity constraint specifications are translated into constraint enforcing mechanisms provided by the DBMS used to implement a database. Most of the commercial DBMSs offer efficient declarative support for the domain constraints, null value constraints, uniqueness constraints and foreign key constraints (key-based IND) [16]. For more complex constraints, using triggers and stored procedures as the procedural mechanisms instead of declarative ones is recommended. Türker and Gertz in [30] emphasize the importance of embedding integrity constraints in the database schema rather then in the application. They state that enforcing integrity constraints and rules identified in the application domain with declarative constraints and/or triggers often is less costly than enforcing the equivalent rules by issuing SQL statements in an application. Preserving of logical data independence is another important reason to embed integrity constraints into database schema. Attaulah and Tompa in [9] stress that the absence of a centralized policy and constraint management system within database systems leads to several problems like the lack of transparency, manageability and compliance of business rules. The approaches presented in [5], [6], [12], [15], [16], [27] and [31] comply with the aforementioned attitudes. We advocate a similar stance and this is an important reason why we develop our SQL Generator to implement the IRICs, besides other integrity constraints.

The growing interest in the Model-Driven Software Development (MDSD) approaches has largely increased the number of tools and methods including code-generation capabilities. Given a platform-independent model (PIM) of an application, these tools generate the application code either by defining an intermediate platform-specific model (PSM) or by executing a direct PIM to code transformation. A conceptual database schema may be seen as a PIM. A transformation of conceptual DB schema into a logical DB schema is a model-to-model (M2M) transformation, while the SQL script generation based on a logical DB schema is a model-to-text (M2T) transformation. Nowadays, almost all tools that support MDSD are able to generate the relational database schemas from PIMs. The major drawback of these tools is that most of them tend to ignore some of the integrity constraints specified in PIMs. Cabot and Teniente in [13] present a survey on the capabilities of current tools regarding the explicit definition of integrity constraints in a PIM and the code generation to enforce them. They classified the different tools in the four categories: CASE tools, MDA (Model-Driven Architecture) specific tools, MDSD tools and OCL (Object Constraint Language) tools. From CASE tools they selected: *Poseidon*, *Rational Rose*, *MagicDraw, Objecteering/UML* and *Together*. In the class of MDA tools *ArcStyler*, *OptimalJ* and *AndroMDA* are evaluated. *OO-Method, WebML* and *Executable UML* are selected beyond MDSD tools, while *Dresden OCL*, *OCLtoSQL*, *OCL2J*, *OCL4Java* and *BoldSoft* are evaluated in the OCL tool class. Most of them do not take the multiplicity constraints into account. The *Objecteering/UML* is an exception to the other tools reviewed in [13], since it allows the use of a trigger system to map the multiplicity constraints, including the minimal multiplicity equal to 1. However, in contrast to our approach, it ignores tuple deletions and updates.

Al-Jumaily, Cuadra and Martinez in [5] present a module to generate triggers for multiplicity constraints verification that is integrated into Rational Rose. In the paper they consider only Oracle DBMS. Although they are tackling similar problem as we are, they are not taking into account mutual dependencies caused by a RIC that exists simultaneously with a considered IRIC. We present the solution of that problem and consider it as the one of the contributions of the paper.

Berrabah and Boufarès in [11] recognize the triggers as a good mean to implement integrity constraints. They distinguish two classes of constraints specified on a UML class diagram: multiplicity and participation constraints. However, furthermore they consider the participation constraints only.

Badaway and Richta in [10] propose an extension to OCL for automatic translation of object level constraints in the modelling language to database level triggers and Zimbrão et al. in [31] proposed a mechanism for translating an OCL constraint to a SQL assertion.

Rybola and Richta in [27] define the multiplicity constraints in a formal way in OCL. They take into consideration both minimal and maximal multiplicity, like we do in our approach. The transformation of OCL specification of constraints into the relational database schema is presented. In the contrast to our approach, the OCL constraints are implemented in SQL as views

selecting records violating the multiplicity restrictions. The authors were motivated with the solutions used in the Dresden OCL toolkit.

Some commercial DBMSs supported triggers before they were covered by the SQL-99 standard. Currently, all major relational DBMS vendors have some support for triggers. However, such support may vary from one to the other DBMS, showing typical deviations from the standard [15]. That is the main reason why we present here the implementation of IRICs for two DBMSs: Oracle 10g and MS SQL Server 2008. They are widely used commercial DBMSs. Besides the similarities, there are significant differences between them in the context of trigger mechanisms. We consider that the examples of IRICs implementation for these two platforms may guide practitioners in solving the similar problems. An automated generation of triggers for IRICs implementation may lead towards less error prone solutions compared to handcrafted database trigger.

Summarizing the related work we may say that we have found just a few approaches tackling the problem of automated implementation of IRICs. Some of them use SQL views to select records violating the multiplicity restrictions. Others use a trigger system, but neither of them consider mutual dependencies caused by a RIC that exists simultaneously with a considered IRIC. Because of that, mechanisms for IRIC's validation require deferred trigger consideration during the transaction. Unfortunately, most of the contemporary DBMSs do not support it and solely use the immediate trigger consideration. Oracle and MS SQL Server have different means that may be used to emulate deferred trigger consideration. In our approach we deal with these differences and suggest possible solutions for both of the DBMSs.

3. Inverse Referential Integrity Constraint

Here we give the definitions of IND, key-based IND, non-key-based IND and IRIC.

Let $N_l(R_l, C_l)$ and $N_r(R_r, C_r)$ be two relation schemes, where N_l and N_r are theirs names, R_l and R_r, corresponding sets of attributes, and C_l and C_r corresponding sets of relation scheme constraints. An inclusion dependency is a statement of the form $N_l[LHS] \subseteq N_r[RHS]$, where *LHS* and *RHS* are non-empty arrays of attributes from R_l and R_r respectively. Having the inclusion operator ($\subseteq$) orientated from the left to right we say that relation scheme N_l is on the left-hand side of the IND, while the relation scheme N_r is on its right-hand side. We use the indexes *l* and *r*, and the names of attribute arrays *LHS* and *RHS*, in order to indicate the left and right hand side of IND, respectively. To define a validation rule of IND we use the following notation: (i) the relation $r(N_l)$ is a set of tuples $u(R_l)$ (or just u) satisfying all constraints from the constraint set C_l; (ii) X-value is a projection of a tuple u on the set of attributes X; and (iii) according to the aforementioned orientation of the inclusion operator, $r(N_l)$ is called the referencing relation, while $r(N_r)$ is called the referenced relation. Informally, a database satisfies the inclusion

dependency if the set of *LHS*-values in the referencing relation $r(N_l)$ is a subset of the set of *RHS*-values in the referenced relation $r(N_r)$.

There are two basic kinds of INDs: key-based INDs and non-key-based INDs. An IND is said to be key-based if *RHS* is a key[1] of the relation scheme N_r. Otherwise, it is a non-key-based. More often key-based IND is called referential integrity constraint. Non-key-based IND with *LHS* that is a key of the relation scheme N_l, where RIC $N_r[RHS] \subseteq N_l[LHS]$ is specified at the same time, is called inverse referential integrity constraint [22]. In Fig. 1 a UML class diagram is used to visually represent this classification of INDs. The associations between the different classes of INDs are also given. A key-based IND, as well as a non-key-based IND, may be seen as a specialization of IND, while an IRIC may be seen as a specialization of a non-key-based IND.

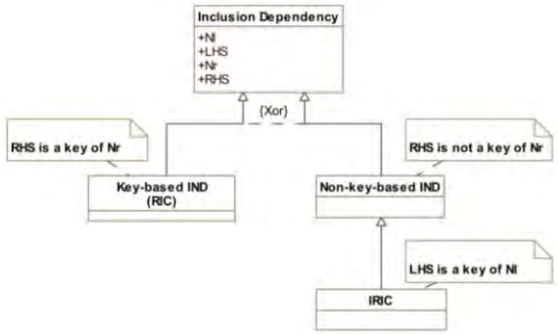


Fig. 1. A classification of different kinds of INDs

Business rules that are to be modelled by the inverse referential integrity constraints often exist in a real world. They are consequences of the mutual existential dependency of the entities of two entity types in a real system.

Example 1. According to the business rules of the university, a department can be established only as a part of a faculty, and a faculty has at least one department. The conceptual database schema expressed by UML class diagram is presented in Fig. 2. The minimal multiplicity of the association *Has* between the class *Faculty* and the class *Department* is one, while the maximal multiplicity is many. After mapping the conceptual DB schema to a

[1] According to [17], a key of a relation scheme $N(R, C)$ is a minimal superkey. Informally, a superkey is any non-empty subset S of R such that no two distinct tuples in any relation $r(N)$ can have the same S-value. In general, a relation scheme may have more than one key (each of them may be called a candidate key). It is common to designate one of them as the primary key. If a relation scheme has only one key it is, at the same time, the primary key of the relation scheme.

relational DB schema according to the transformation rules suggested in [17] we produce a relational database schema containing two relation schemes: *Faculty* and *Department*, with the keys *FacId* and *FacId+DepId* respectively, and two inclusion dependencies *IND1* and *IND2*:

Faculty({*FacId, FacShortName, FacName, Dean*}, {*FacId*}),
Department({*FacId, DepId, DepName*}, {*FacId+DepId*}),
IND1: Department [*FacId*] $\subseteq$ *Faculty* [*FacId*],
IND2: Faculty[*FacId*] $\subseteq$ *Department*[*FacId*].

Since that *FacId* is the key of relation scheme *Faculty*, *IND1* is the key-based inclusion dependency, i.e. the referential integrity constraint. It is modelling the business rule that a department can be established only as a part of a faculty. The constraint *IND2* is the non-key-based inclusion dependency, since that *FacId* is not the key of relation scheme *Department*. The *FacId* is the key of the relation scheme *Faculty*, which is on the left-hand side of the inclusion dependency's specification and the referential integrity constraint *IND1* is specified as well. Therefore, the constraint *IND2* is the inverse referential integrity constraint. It is modelling the business rule that faculty must have at least one department. □

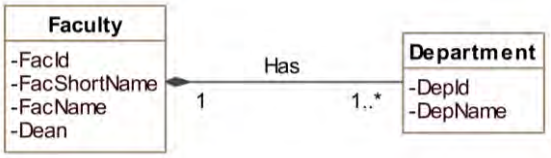


Fig. 2. A university conceptual database schema

Programmers are obliged to manage IRICs via procedural mechanisms (procedures and triggers). That is the reason why the IRICs are mostly implemented at the middle layer instead at the DB server. Still, the validation of the IRICs at the DB server: (i) cuts the costs of the application maintaining; (ii) provides better performances due to the less traffic in the typical client-server architecture; (iii) enables the same way of preventing the violation of a database consistency.

In this paper the methods for the implementation of IRICs, using the mechanisms provided by relational database systems are presented. These methods are implemented in the SQL Generator that provides creating SQL scripts according to the syntax of: (i) ANSI SQL:2003 standard [8], (ii) DBMS Microsoft (MS) SQL Server 2008 with MS T-SQL [21], and (iii) DBMS Oracle 10g with Oracle PL/SQL [24]. In the context of the approach presented in this paper, there are no crucial syntax differences in SQL languages of the DBMSs used in this paper, in comparison to the newer releases of the same DBMSs, i.e. MS SQL Server 2012 and Oracle 11g, respectively. Therefore, considerations given in this paper may be applied also at these DBMSs, without any limits.

4. Algorithms for IRIC Validation

By specifying the IRIC $N_j[Y] \subseteq N_i[X]$ it comes towards the bogus mutual insertion blocking of the instances of the relation schemes N_i and N_j, since the RIC $N_i[X] \subseteq N_j[Y]$ is also specified. The notion „mutual insertion blocking" is used to illustrate the following situation: (i) it is not possible to insert a new tuple into the relation $r(N_i)$ with not null values for all attributes $A \in X$, unless there is a tuple in the relation $r(N_j)$ with the Y value same as the X value of the inserted tuple (due to the RIC $N_i[X] \subseteq N_j[Y]$); and, also (ii) it is not possible to insert a new tuple into the relation $r(N_j)$ with a Y value given, unless there is a tuple in the relation $r(N_i)$ with the X value same as the aforementioned Y value (due to the IRIC $N_j[Y] \subseteq N_i[X]$) [23].

Example 2. In Fig. 3 it is presented a database instance of the database schema from Example 1. Due to the existence of the referential integrity *IND1* it is not possible to insert the tuple (2, D2, 'Dentistry') into the relation *Department*. However, due to the specified inverse referential integrity *IND2* it is even not possible to insert the tuple (2, 'FOM', 'Faculty of Medicine', 'Simpson') into the relation *Faculty*. These tuples are said to be mutually blocked. □

Faculty

FacId	*FacShortName*	*FacName*	*Dean*
1	MAT	Mathematics	Smith

Department

FacId	*DepId*	*DepName*
1	D1	Geometry

Fig. 3. A University database instance

The algorithms for insertion, deletion and modification control in the presence of inverse referential integrity constraints are presented in Fig. 4, Fig. 5 and Fig. 6, respectively.

Apart from the notation already introduced at the beginning of Section 3, in the algorithms we use the following notation: $u[X]$ denotes X-value of a tuple u, $|X|$ denotes the cardinality of an array of attributes X, ω denotes the null value, and $K_p(R_i)$ denotes the primary key of a relation scheme N_i.

In the following text these algorithms are described in more details.

Let $N_j[Y] \subseteq N_i[X]$ is an IRIC. In the context of the IRIC, $r(N_j)$ is the referencing relation, while $r(N_i)$ is the referenced relation, since the relation scheme N_j is on the left-hand side and the relation scheme N_i is on the right-hand side of the IRIC. The IRIC may be violated in three cases: (i) when a tuple is inserted into the referencing relation, (ii) when a tuple is deleted from the referenced relation or (iii) when a tuple's X-value is modified in the referenced relation.

Trigger:	INSERTION CONTROL IN THE PRESENCE OF IRICs
Definition area: **Relation schemes**: N_i, N_j **Attributes:** $X = (A_1, ..., A_{\|X\|})$, $Y = (B_1, ..., B_{\|Y\|})$ $\|X\| = \|Y\| \wedge (\forall\, l \in \{1, \ldots, \|X\|\})(dom(A_l) \subseteq dom(B_l) \wedge A_l \in R_i \wedge B_l \in R_j)$	
Specification of the constraint: i: $N_i[Y] \subseteq N_i[X]$	
Specification of the operation: **Time**: AFTER OPERATION **Operation**: INSERT	
Data Inputs	
From DB	$r(N_i)$, $r(N_j)$
Input	v: tuple that would be inserted into $r(N_j)$
Local declarations:*ind* (*ind* = 1 – constraint is satisfied, *ind* = 0 – constraint is violated)	

Pseudo code:

```
BEGIN PROCESS  Insert_inv_ref_int
  SET  ind ← 0
  FOR ALL u∈r(Ni) DO        // Search in the relation r(Ni) for v[Y] value
    IF  v[Y] = u[X]  THEN
      SET  ind ← 1
      BREAK
    ENDIF
  ENDFOR
  IF  ind = 0  THEN
    CANCEL_OPERATION('Error description')
  ENDIF
ENDPROCESS  Insert_inv_ref_int
```

Fig. 4. An algorithm for insertion control

An algorithm for the control of insertions (Fig. 4) will reject the insert operation of the *v* tuple into the referencing relation if the referenced relation doesn't contain any tuple with *X*-value matching the *Y*-value of the tuple *v*.

Trigger:	DELETION CONTROL IN THE PRESENCE OF IRICs
Definition area: **Relation schemes**: N_i, N_j **Attributes:** $X = (A_1, ..., A_{\|X\|})$, $Y = (B_1, ..., B_{\|Y\|})$ $\|X\| = \|Y\| \wedge (\forall I \in \{1, \ldots, \|X\|\})(dom(A_I) \subseteq dom(B_I) \wedge A_I \in R_i \wedge B_I \in R_j)$	
Specification of the constraint: i: $N_j[Y] \subseteq N_i[X]$	
Specification of the operation: **Time**: AFTER OPERATION **Operation**: DELETE	
Data Inputs	
From DB	$r(N_i)$, $r(N_j)$
Input	u: tuple that would be deleted from $r(N_i)$
Local declarations:*ind* (*ind* = 1 – constraint is satisfied, *ind* = 0 – constraint is violated)	

Pseudo code:

```
BEGIN PROCESS Delete_inv_ref_int
  SET ind ← 0
  FOR ALL A∈X DO          // Search for null value in X-value
    IF u[A] = ω THEN
      SET ind ← 1
      BREAK
    ENDIF
  ENDFOR
  IF ind = 0 THEN
   FOR ALL t∈r(Ni) DO      // Search in r(Ni)
     IF t[Kp(Ri)] ≠ u[Kp(Ri)] ∧ u[X] = t[X] THEN
       SET ind ← 1
       BREAK
     ENDIF
   ENDFOR
  ENDIF
  IF ind = 0 THEN
    EXECUTE ACTIVITY
  ENDIF
ENDPROCESS Delete_inv_ref_int
```

Fig. 5. An algorithm for deletion control

Trigger:	MODIFICATION CONTROL IN THE PRESENCE OF IRICs
Definition area: **Relation schemes**: N_i, N_j **Attributes:** $X = (A_1, ..., A_{\|X\|})$, $Y = (B_1, ..., B_{\|Y\|})$ $\|X\| = \|Y\| \wedge (\forall l \in \{1, \ldots, \|X\|\})(dom(A_l) \subseteq dom(B_l) \wedge A_l \in R_i \wedge B_l \in R_j)$	
Specification of the constraint: i: $N_j[Y] \subseteq N_i[X]$	
Specification of the operation: **Time**: AFTER OPERATION **Operation**: UPDATE	
Data Inputs	
From DB	$r(N_i)$, $r(N_j)$
Input	u: original tuple to be modified in $r(N_i)$ u': new tuple obtained by the modification of tuple u
Local declarations:*ind* (*ind* = 1 – constraint is satisfied, *ind* = 0 – constraint is violated)	

Pseudo code:

```
BEGIN PROCESS Update_inv_ref_int
  IF u'[X] ≠ u[X] THEN
    SET ind ← 0
    FOR ALL A∈X DO        // Search for null value in X-value
       IF u[A] = ω THEN
        SET ind ← 1
         BREAK
       ENDIF
    ENDFOR
    IF ind = 0 THEN
      FOR ALL t∈r(Ni) DO    // Search in r(Ni)
        IF t[Kp(Ri)] ≠ u[Kp(Ri)] ∧ u[X] = t[X] THEN
          SET ind ← 1
           BREAK
        ENDIF
       ENDFOR
    ENDIF
    IF ind = 0 THEN
      CANCEL_OPERATION('Error description')
    ENDIF
  ENDIF
ENDPROCESS Update_inv_ref_int
```

Fig. 6. An algorithm for modification control

An algorithm for the control of deletions (Fig. 5) detects an IRIC's violation when a tuple *u* from the referenced relation is deleted and if the conjunction of conditions is satisfied: (i) *X*-value of the tuple *u* doesn't contain null values; and (ii) the referenced relation doesn't contain another tuple *t* (strictly different from the tuple *u*) with *X*-value matching the *X*-value of the tuple *u*. The first condition needs additional explanation. Namely, *Y* is the key for the left-hand side relation scheme. Consequently, neither of the tuples from the referencing relation can contain null value in the *Y*-value sequence. Therefore, neither of the tuples from the referenced relation that contains null values can be referenced by some tuple from referencing relation. It may be concluded that by the deletion of such a tuple from $r(N_i)$, IRIC cannot be violated. If a constraint violation is detected, the algorithm will reject the delete operation or, alternatively it will delete all tuples from the referencing relation having the *Y*-value matching the *X*-value of the tuple *u*. During the IRIC implementation pseudo-instruction *EXECUTE ACTIVITY* will be replaced with an appropriate program code for the selected action.

An algorithm for the control of modifications (Fig. 6) will reject the update operation of the tuple *u* from the referenced relation if the conjunction of conditions is satisfied: (i) the update operation changes the tuple's *X*-value ($u'[X] \neq u[X]$, where *u* is the tuple before the modification and *u'* is the tuple after the modification); (ii) the original *X*-value (*X*-value of the tuple *u* before the modification) doesn't contain null values; and (iii) the referenced relation doesn't contain any other tuple *t* (strictly different from the original tuple *u*) with *X*-value matching the original *X*-value. The explanation for the second condition is analogous to the explanation for the first condition in the previous paragraph.

5. Implementation of IRICs by Procedural Mechanisms

DBMSs have different mechanisms for the implementation of relational database constraints (RDBCs). We are going to classify these mechanisms into two categories. The *core* mechanisms use the CONSTRAINT clause within the CREATE / ALTER TABLE statements of SQL, to implement a RDBC. The *additional* mechanisms use the CREATE ASSERTION statement or the CREATE TRIGGER statement to implement a RDBC. The fundamental mechanisms are declarative, while the additional mechanisms may be declarative (e.g. assertions) or procedural (e.g. triggers). An IRIC cannot be implemented by means of core mechanisms of contemporary DBMSs. Therefore, it has to be implemented via declarative assertions or procedural triggers. Albeit SQL standards allow assertions, most of the contemporary DBMSs do not support them. Therefore, we have to implement the IRICs via DBMS procedural mechanisms, by creating triggers, alongside with the required functions and procedures. Another problem to be solved occurs due to a mutual insertion blocking, caused by an IRIC specification. Because of that, mechanisms for IRIC's validation require deferred trigger

consideration during the transaction. Some DBMSs support the deferred constraint consideration. Unfortunately, most of the contemporary DBMSs do not support deferred trigger consideration and provide the immediate trigger consideration only.

Our SQL Generator enables an automated implementation of the IRICs for DBMSs MS SQL Server 2008 [21] and Oracle 10g [24]. One of the reasons for their selection is that they are widely used commercial DBMSs. Another reason is that besides the similarities, there are significant differences between them.

In the context of IRICs implementation the main similarities are that: (i) both of them do not support assertions and deferred trigger consideration; and (ii) there are many similarities concerning trigger specification. The major difference in the same context is that Oracle and MS SQL Server have different means that may be used to emulate deferred trigger consideration. Oracle enables global variables declaration in packages. We can use them to pass the information that a trigger has to skip an IRIC checking. The global variables can't be declared in MS SQL Server. Instead, we use tuple in auxiliary table to pass the information that a trigger has to skip an IRIC checking. In this section we will illustrate the differences between IRICs' implementation techniques for MS SQL Server 2008 and Oracle 10g, used in our SQL Generator.

In our approach the procedural implementation of a constraint, can be unified. It consists of the following steps: (i) specifying a parameterized pattern of the algorithm for a specific DBMS, (ii) replacing the pattern parameters with real values, and (iii) generating an SQL script comprising necessary triggers, procedures and functions [1].

5.1. IRIC Implementation for MS SQL Server 2008

In this section, parameterized patterns of the algorithms for controlling the IRIC validation during the insert, update and delete operations for DBMS MS SQL Server 2008 (MS SQL) are given.

5.1.1. Patterns for tuple insertion in the presence of an IRIC

In order to keep the DB consistency in the presence of the IRICs, mutually blocked tuples (like those in Example 2) must be inserted in one transaction. There are two ways to do that: (i) a view created over the relations $r(N_i)$ and $r(N_j)$ may be used for the double insertion; or (ii) a custom DB procedure for double insertion may be developed.

The pattern of the trigger using views for tuple insertion is presented in Fig. 7. For each specified IRIC $N_j[Y] \subseteq N_i[X]$ the trigger based on that pattern would be generated. We use a special MS SQL Server table, named *Inserted*, that stores copies of the affected tuples during the execution of INSERT and UPDATE statements. The values of attributes from R_j and R_i are

separated and two INSERT statements are specified for tuple insertion into relations $r(N_j)$ and $r(N_i)$, respectively. Since these tuples are mutually blocked, a trigger for tuple insertion in $r(N_j)$ (Fig. 10), will raise the error. To prevent that, we emulate the deferred trigger consideration, using an auxiliary DB relation *Trigger_Stat*. The tuple with given trigger name and transaction ID is to be written in the *Trigger_Stat* relation, by calling the procedure *Trigger_Ex* (Fig. 8) with 0 as the first argument. This tuple is aimed to pass the information that the trigger for tuple insertion in $r(N_j)$ (Fig. 10) has to skip an IRIC checking. In that way we emulate the deferred trigger consideration. Thus, the tuple insertion in $r(N_j)$ is enabled. Afterwards, the insertion of corresponding tuple in $r(N_i)$ is allowed, since it does not violate the RIC $N_i[X] \subseteq N_j[Y]$ any more. The next step is to re-enable IRIC checking in the trigger for tuple insertion in $r(N_j)$ (Fig. 10). In order to do it, the previously inserted tuple in the *Trigger_Stat* relation, with given trigger name and transaction ID, is to be deleted, by calling the procedure *Trigger_Ex* (Fig. 8) with 1 as the first argument. At the end, the function *ContainmentIRI_*<N_j> (Fig. 9) is called in order to check if the IRIC $N_j[Y] \subseteq N_i[X]$ is violated.

```
CREATE TRIGGER TRG_<Const_Name>_View
ON View_<Nj>_<Ni> INSTEAD OF INSERT
AS
  DECLARE @Idt int, @Count int, <Decl_Var_For_Ni_Nj>
  SELECT <Var_array_For_Ni_Nj> FROM Inserted
  SET @Idt = @@SPID
  exec dbo.Trigger_Ex 0, 'WriteRI_<Nj>', @Idt
  INSERT INTO <Nj> VALUES (<Var_array_For_Nj>)
  INSERT INTO <Ni> VALUES (<Var_array_For_Ni>)
  exec dbo.Trigger_Ex 1, 'WriteRI_<Nj>', @Idt
  IF dbo.ContainmentIRI_<Nj>(<Var_For_Y>) = 0
  BEGIN
     RAISERROR('IRIC violation!',16,1)
     ROLLBACK TRAN
  END
```

Fig. 7. A pattern of the trigger over view

The trigger generator creates the trigger from the pattern presented in Fig. 7. by replacing:

- <N_j> with the name of relation scheme N_j;
- <N_i> with the name of relation scheme N_i;
- <*Const_Name*> with IRI_<N_j>_<N_i>, where IRI marks that it is the inverse referential integrity constraint, and <N_j> and <N_i> will be replaced with the names of relation schemes N_j and N_i respectively;
- <*Decl_Var_For_Ni_Nj*> with the list of variable declarations of the form @<*Attribute_Name*> *data type*, for each attribute from R_i and R_j;
- <*Var_array_For_Ni_Nj*> with the list of variables declared by the list of declarations that replaced <*Decl_Var_For_Ni_Nj*>. These variables are

set to appropriate values from a tuple contained in the MS SQL Server table *Inserted*;
- <*Var_array_For_Nj*> and <*Var_array_For_Ni*> with the lists of variables containing the input value for each attribute from R_j and R_i, respectively; and
- <*Var_For_Y*> with the list of variables declared by the list of declarations that replaced <*Decl_Var_For_Ni_Nj*>, containing only those variables that are related to the attributes from *Y*. List elements are of the form @<*Name_of_Attribute_From_Y*>. The list of variables represents the argument of function *ContainmentIRI_*<N_j>.

The procedure *Trigger_Ex* and the pattern of the function *ContainmentIRI_*<N_j>, that are called from a trigger based on the pattern in Fig. 7, are presented in Fig. 8 and Fig. 9 respectively.

```
CREATE PROCEDURE dbo.Trigger_Ex
(@Stat int, @Trigger_Name varchar(50), @Idt int)
AS
  IF @Stat = 1
    DELETE FROM Trigger_Stat WHERE
    Trigger = @Trigger_Name AND IdTransaction = @Idt
  ELSE
    INSERT INTO Trigger_Stat (Trigger, IdTransaction)
    VALUES (@Trigger_Name, @Idt)
```

Fig. 8. A SQL procedure for trigger execution control

The procedure *Trigger_Ex* in Fig. 8 is used in the process of trigger's execution control. In the suggested solution an auxiliary DB relation *Trigger_Stat* is used, as we already explained, earlier in this section.

```
CREATE FUNCTION dbo.ContainmentIRI_<Nj>(<Decl_Var_For_Y>)
RETURNS int
AS
  BEGIN
    DECLARE @Count int, @Ret  int
    SELECT @Count = COUNT(*) FROM <Ni> u
    WHERE (<Selection_Cond>)
    IF @Count != 0
      SELECT @Ret =1
    ELSE
      SELECT @Ret =0
    RETURN @Ret
  END
```

Fig. 9. A pattern of the *ContainmentIRI_*<N_j> function

A DB function *ContainmentIRI_*<N_j> is to be called from a trigger based on the pattern in Fig. 7. It is generated from the function pattern in Fig. 9, by replacing:
- <N_j> with the name of relation scheme N_j;

- <*Decl_Var_For_Y*> with the list of parameter declarations of the form @<*Name_of_Attribute_From _Y*> *data type*, for each attribute from *Y*; and
- <*Selection_Cond*> with a conjunction of relational expressions of the form:
  u.<*Name_of_Attribute_From _X*> = @<*Name_of_Attribute_From _Y*>.

SQL code for view creation is trivial, and therefore it is omitted here. We only emphasize that it should contain all attributes from both R_i and R_j.

In order to prevent the IRIC violation due to the separate insertion of mutually blocked tuples, a trigger adhering the pattern in Fig. 10 is created. The replacement of the parameters during the trigger generation is analogous to the replacement of corresponding parameters in patterns from figures 7, 8 and 9.

Finally, the SQL function for trigger execution is presented in Fig. 11. This function is aimed to detect if there is a tuple, with given trigger name and transaction ID, in the auxiliary table *Trigger_Stat*. It is called from a trigger for tuple insertion in $r(N_j)$ (Fig. 10). The return value 0 (a tuple exists) indicates that IRIC check is to be done, while the return value 1 (a tuple doesn't exist) indicates that it is to be skipped.

```
CREATE TRIGGER TRG_<Nj>_<Const_Name>_INS
ON <Nj> FOR INSERT
AS
  IF dbo.ExecuteTrigger(TRG_<Nj>_<Const_Name>_INS)=0
  BEGIN
    RAISERROR('Data have to be inserted via view:
        View_<Nj>_<Ni> or procedure Insert_<Const_Name>',16,1)
    ROLLBACK TRAN
  END
```

Fig. 10. A tuple insertion control pattern

```
CREATE FUNCTION dbo.ExecuteTrigger(@Trigger_Name varchar(50))
RETURNS int
AS
  BEGIN
    DECLARE @Count int, @Idt  int, @Ret int
    SELECT @Idt = @@SPID
    SELECT @Count = COUNT(*) FROM Trigger_Stat
    WHERE (Trigger = @Trigger_Name) AND (IdTransaction = @Idt)
    IF @Count != 0
      SELECT @Ret =1
    ELSE
      SELECT @Ret =0
    RETURN @Ret
  END
```

Fig. 11. A SQL function for trigger execution control

```
CREATE PROCEDURE dbo.Insert_<Const_Name>
(<Decl_Var_For_Ni_Nj>)
AS
  DECLARE @Idt int
  BEGIN TRANSACTION
    SET @Idt = @@SPID
    exec dbo. Trigger_Ex 0, 'WriteRI_<Nj>', @Idt
    INSERT INTO <Nj> VALUES (<Var_array_For_Nj>)
    INSERT INTO <Ni> VALUES (<Var_array_For_Ni>)
    exec dbo.Trigger_Ex 1, 'WriteRI_<Nj>', @Idt
    IF dbo.ContainmentIRI_<Nj>(<Var_For_Y>) = 0
    BEGIN
      RAISERROR('IRIC violation!',16,1)
      ROLLBACK TRAN
    END
  COMMIT TRANSACTION
```

Fig. 12. A pattern of the procedure for tuple insertion

Another way for providing insertion of mutually blocked tuples in one transaction is by creating a custom DB procedure for double insertion. Parameterized pattern for such a procedure is given in Fig. 12. The meaning of parameters is similar to that in the pattern of the trigger for double insertion using views, and therefore is omitted here.

5.1.2. Patterns for tuple deletion in the presence of an IRIC

The pattern of the trigger for tuple deletion is presented in Fig. 13. For each specified IRIC $N_j[Y] \subseteq N_i[X]$ the trigger based on that pattern would be generated. The trigger generator creates the trigger from the pattern by replacing:

- <N_i> with the name of relation scheme N_i;
- <*Const_Name*> with IRI_<N_j>_<N_i>, where IRI marks that it is the inverse referential integrity constraint, and <N_j> and <N_i> will be replaced with the names of relation schemes N_j and N_i respectively;
- <*Decl_Var_For_X*> with the list of variable declarations of the form @<*Attribute_Name*> *data type*, for each attribute from X;
- <*Attr_from_X*> with the list containing the names of attributes from X;
- <*Condition*> with the conjunction of relational expressions of the form:

  @<*Name_of_Attribute_From _X*> IS NOT NULL;

- <*Selection_Cond*> with the conjunction of relational expressions of the form:

  u.<*Name_of_Attribute_From_X*> = @<*Name_of_Attribute_From_X*>; and

- <*Var_For_X*> with the list of variables' names declared by the list of declarations that replaced <*Decl_Var_For_X*>. List elements are of the form @<*Name_of_Attribute_From_X*>.

```
CREATE TRIGGER TRG_<Ni>_<Const_Name>_DEL
ON <Ni> FOR DELETE
AS
  DECLARE @Count int, <Decl_Var_For_X>
  DECLARE Cursor_<Ni> CURSOR
  FOR SELECT <Attr_From_X> FROM Deleted
  OPEN Cursor_<Ni>
  FETCH NEXT FROM Cursor_<Ni> INTO <Var_For_X>
  WHILE @@FETCH_STATUS=0
  BEGIN
    IF <Condition>
    BEGIN
      SELECT @Count = COUNT(*) FROM <Ni> u
      WHERE (<Selection_Cond>)
      IF @Count = 0
          <Execute_Activity>
    END
    FETCH NEXT FROM Cursor_<Ni> INTO <Var_For_X>
  END
  CLOSE Cursor_<Ni>
  DEALLOCATE Cursor_<Ni>
```

Fig. 13. A pattern of the delete trigger

Deleted table, used in a declaration of cursor *Cursor_*<N_i>, is a special MS SQL Server table that stores copies of the affected tuples during the execution of DELETE and UPDATE statements. Depending on the selected activity, <*Execute_Activity*> is replaced with *CascadeIRI_Del_*<N_i> (Fig. 14) procedure call (Cascade delete) or with SQL code for activity restriction (Fig. 15).

The procedure generator creates the procedure for cascade deletion (Fig. 14) from the pattern by replacing:

- <N_i> with the name of relation scheme N_i;
- <*Decl_Var_For_X*> with the list of variable declarations of the form @<*Attribute_Name*> *data type*, for each attribute from *X*; and
- <*Selection_Cond*> with the conjunction of relational expressions: *v.*<*Name_of_Attribute_From_Y*> = @<*Name_of_Attribute_From_X*>.

```
CREATE PROCEDURE dbo.CascadeIRI_Del_<Ni>(<Decl_Var_For_X>)
AS
  DELETE FROM <Ni> v WHERE (<Selection_Cond>)
```

Fig. 14. A pattern of procedure for cascade deletion

```
BEGIN
  RAISERROR('The tuple from relation <Ni> could not be deleted',16,1)
  ROLLBACK TRAN
END
```

Fig. 15. A pattern of SQL code for operation restriction

5.1.3. Patterns for tuple modification in the presence of an IRIC

The pattern of the trigger for tuple modification for MS SQL Server is presented in Fig. 16.

```
CREATE TRIGGER TRG_<Ni>_<Const_Name>_UPD
ON <Ni> FOR UPDATE
AS
  DECLARE @Count int, <Decl_Var_For_X>
  IF <Modification_Cond>
  BEGIN
    DECLARE Cursor_<Ni> CURSOR
    FOR SELECT <Attr_From_X> FROM Deleted
    OPEN Cursor_<Ni>
    FETCH NEXT FROM Cursor_<Ni> INTO <Var_For_X>
     WHILE @@FETCH_STATUS=0
     BEGIN
       IF <Condition>
       BEGIN
         SELECT @Count = COUNT(*) FROM <Ni> u
         WHERE (<Selection_Cond>)
         IF @Count = 0
         BEGIN
           RAISERROR('The tuple from relation <Ni> could not be updated',16,1)
           ROLLBACK TRAN
         END
       END
       FETCH NEXT FROM Cursor_<Ni> INTO <Var_For_X>
     END
     CLOSE Cursor_<Ni>
     DEALLOCATE Cursor_<Ni>
  END
```

Fig. 16. A pattern of the modification trigger

The replacement of the parameters is same as the replacement of parameters for the deletion trigger. The modification trigger has one more parameter, <*Modification_Cond*>. During the trigger generation it is replaced by the disjunction of SQL functions UPDATE(<*Name_of_Attribute_from_X*>) for each attribute belonging to the attribute set *X*.

5.2. IRIC Implementation for Oracle 10g

Existence of the SQL standard may be considered as one of the major reasons for the commercial success of relational databases. The RDBMSs' vendors make efforts to achieve high SQL standard compliance. Despite this, in practice, there are many differences between various RDBMSs. In the context of IRICs implementation, the differences concerning the means that may be used to emulate deferred trigger consideration are crucial. The global variables can't be declared in MS SQL Server. Instead, we use a tuple in auxiliary table to pass the information that a trigger has to skip an IRIC checking, as it is shown in Section 5.1. Oracle DBMS enables global variables declaration in packages. They can be used to pass the information that a trigger has to skip an IRIC checking. Therefore, here we present the parameterized patterns for triggers and procedures implementing algorithms from Section 4, for Oracle DBMS. Some of parameters in the patterns for Oracle DBMS are same as the parameters in the corresponding patterns for MS SQL Server. The explanation of their replacement will be omitted in the following text. The replacement of the parameters those are specific for patterns for Oracle Server will be explained in details.

5.2.1. Patterns for tuple insertion in the presence of an IRIC

As well as for MS SQL Server, there are two ways to insert mutually blocked tuples in one transaction: (i) a view created over the relations $r(N_i)$ and $r(N_j)$ may be used for the double insertion; or (ii) a custom DB procedure for double insertion may be developed. The pattern of the trigger using views for tuple insertion is presented in Fig. 17.

Here we notify the basic differences between the patterns for tuple insertion in the presence of an IRIC for MS SQL Server and Oracle. The MS SQL Server has the *Inserted* table that stores copies of the affected tuples during the execution of INSERT and UPDATE statements. In Oracle notation key-words NEW and OLD are used for that purpose. NEW is used to refer to a newly inserted or newly updated tuple. OLD is used to refer to a deleted tuple or to a tuple before it was updated. The values of attributes from R_j and R_i are separated and two INSERT statements are specified for tuple insertion into relations $r(N_j)$ and $r(N_i)$, respectively. Since these tuples are mutually blocked, a trigger for tuple insertion in $r(N_j)$ (Fig. 20), will raise an error. To prevent that, we need to emulate the deferred trigger consideration. Unlike MS SQL Server, Oracle enables global variables declaration in packages. So, in a package (Fig. 18) the global variable *Trigger_Ex* is declared. Before the first INSERT statement in Fig. 17, the *Trigger_Ex* is set to FALSE, indicating that a trigger for tuple insertion in $r(N_j)$ (Fig. 20) has to skip an IRIC checking. In that way we emulate the deferred trigger consideration. Thus, the tuple insertion in $r(N_j)$ is enabled, without raising an application error. Afterwards, the insertion of corresponding tuple in $r(N_i)$ is allowed, since it

does not violate the RIC $N_i[X] \subseteq N_j[Y]$ any more. The next step is to re-enable IRIC checking in the trigger for tuple insertion in $r(N_j)$ (Fig. 20). In order to do it in Oracle, the *Trigger_Ex* is set to TRUE, indicating that a trigger for tuple insertion in $r(N_j)$ (Fig. 20) has to enforce an IRIC checking. At the end, the function *ContainmentIRI_*<N_j> (Fig. 19) is called in order to check if the IRIC $N_j[Y] \subseteq N_i[X]$ is violated.

```
CREATE OR REPLACE TRIGGER TRG_<Const_Name>_View
INSTEAD OF INSERT ON View_<Nj>_<Ni>
FOR EACH ROW
DECLARE
  I NUMBER;
  Exc EXCEPTION;
  t <Nj>%ROWTYPE;
BEGIN
  SELECT COUNT(*) INTO I FROM <Ni> WHERE (<Selection_Cond>);
  IF I <> 0 THEN
    INSERT INTO <Ni> VALUES (<Attr_Value_From_Nj>);
  ELSE
    <Const_Name>_PCK.Trigger_Ex := FALSE;
    INSERT INTO <Nj> VALUES (<Attr_Value_From_Nj>);
    INSERT INTO <Ni> VALUES (<Attr_Value_From_Ni>);
    <Const_Name>_PCK.Trigger_Ex := TRUE;
    SELECT * INTO t
    FROM <Nj> WHERE (<Selection_Cond>);
    IF NOT Global_PCK.ContainmentIRI_<Nj>(t) THEN
      RAISE Exc;
    END IF;
  END IF:
  EXCEPTION WHEN Exc THEN
    RAISE_APPLICATION_ERROR (-20001,'IRIC violation!');
END;
```

Fig. 17. A pattern of the trigger over view for Oracle

The replacement of parameters, specific for Oracle patterns, during the trigger generation is done as follows:

- <*Selection_Cond*> is replaced by the conjunction of relational expressions (one expression per each attribute from *Y*) of the form:

<*Name_of_Attribute_From _Y*> = :NEW.<*Name_of_Attribute_From _Y*>;

- <*Attr_Value_From_Ni*> (<*Attr_Value_From_Nj*>) is replaced by the list of elements (one element per each attribute from R_i or R_j) of the form:

:NEW.<*Name_of_Attribute_From _Ni*>

(:NEW.<*Name_of_Attribute_From _Nj*>).

Trigger_Ex is a global variable defined in a package created for the appropriate constraint. The variable gets value TRUE if the trigger ought to be executed and gets value FALSE otherwise. The parameterized content of

that package is presented in Fig. 18. The package parameter <*Attr_Decl_Rec_X*> is replaced with the list of elements of the form:

<*N*$_i$>.<*Name_of_Attribute_From_X*>*%TYPE*.

*For_*<*N*$_i$> and *Count_IRI* are variables declared in the package presented in Fig. 18. They are used in modification and deletion triggers, and will be explained in Section 5.2.2.

```
CREATE OR REPLACE PACKAGE <Const_Name>_PCK
IS
  TYPE TRec<Ni> IS RECORD (<Attr_Decl_Rec_X>);
  TYPE TTabForDelUpd IS TABLE OF TRec<Ni> INDEX BY BINARY_INTEGER;
  For_<Ni> TTabForDelUpd;
  Count_IRI NUMBER(8,0);
  Trigger_Ex BOOLEAN;
END;
```

Fig. 18. A pattern of IRIC's package

```
FUNCTION ContainmentIRI_<Nj> (v IN <Nj>%ROWTYPE)
RETURN BOOLEAN
IS
  I  NUMBER;
BEGIN
  SELECT COUNT(*) INTO I FROM <Ni> u
  WHERE (<Selection_Cond>);
  IF I <> 0 THEN
    RETURN TRUE;
  ELSE
    RETURN FALSE;
  END IF;
END;
```

Fig. 19. A pattern of the *ContainmentIRI_*<N$_j$> function

The pattern of the DB function *ContainmentIRI_*<*N*$_j$>, called from *TRG_*<*Const_Name*>*_View* trigger (Fig. 17) is shown in Fig. 19. The function is to be defined in global package *Global_PCK*. During the function generation process the parameter <*Selection_Cond*> is replaced by the conjunction of relational expressions (one expression per each attribute from *Y*) of the form:

u.<*Name_of_Attribute_From_X*> = *v*.<*Name_of_Attribute_From_Y*>.

In order to prevent the IRIC violation due to the separate insertion of mutually blocked tuples, a trigger adhering to the pattern in Fig. 20 is to be created.

```
CREATE OR REPLACE TRIGGER TRG_<Nj>_<Const_Name>_INS
BEFORE INSERT ON <Nj> FOR EACH ROW
BEGIN
  IF <Const_Name>_PCK.Trigger_Ex = TRUE THEN
    RAISE_APPLICATION_ERROR(-20004, 'Data have to
      be inserted via view:View_<Nj>_<Ni> or procedure Insert_<Const_Name>');
  END IF;
END;
```

Fig. 20. A tuple insertion control pattern

```
CREATE OR REPLACE PROCEDURE Insert_<Const_Name>
(v IN <Nj>%ROWTYPE, u IN <Ni>%ROWTYPE)
IS
  t <Nj>%ROWTYPE;
  Exc EXCEPTION;
BEGIN
  <Const_Name>_PCK.Trigger_Ex := FALSE;
  INSERT INTO <Nj> VALUES (<Attr_Value_From_Nj>);
  INSERT INTO <Ni> VALUES (<Attr_Value_From_Ni>);
  <Const_Name>_PCK.Trigger_Ex := TRUE;
   SELECT * INTO t
  FROM <Nj> WHERE (<Selection_Cond>);
  IF NOT Global_PCK.ContainmentIRI_<Nj>(t) THEN
     RAISE Exc;
  END IF;
  EXCEPTION WHEN Exc THEN
    RAISE_APPLICATION_ERROR (-20001,'IRIC violation!');
END;
```

Fig. 21. A pattern of the procedure for mutually blocked tuples insertion

Another way for providing insertion of mutually blocked tuples in one transaction is by creating a custom DB procedure for double insertion. Parameterized pattern for such a procedure is given in Fig. 21. The differences between a procedure for mutually blocked tuples insertion for MS SQL Server (see Fig. 12) and appropriate procedure for Oracle (see Fig. 21), are the same as the differences described at the beginning of this section (see Fig. 7 and Fig. 17). The meaning of parameters is similar to that in the pattern of the trigger for double insertion using views, and therefore is omitted here.

5.2.2. Patterns for tuple deletion in the presence of an IRIC for Oracle

Here we notify basic differences between the patterns for tuple deletion in the presence of an IRIC for MS SQL Server and Oracle. MS SQL Server provides the *Deleted* table that stores copies of the affected tuples during the execution of DELETE and UPDATE statements. In Oracle notation key-words

NEW and OLD are used for that purpose. Hereof, for Oracle 10g three triggers are to be created for the implementation of tuple deletion under the presence of IRICs. The first one is run at the statement level, before the tuple deletion. It has an assignment to set the auxiliary data structures, used by other triggers. The pattern for the first trigger is shown in Fig. 22.

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_DEL1
BEFORE DELETE <Ni>
BEGIN
  <Const_Name>_PCK.Count_IRI := 0;
  <Const_Name>_PCK.For_<Ni>.DELETE;
END;
```

Fig. 22. A pattern of the first delete trigger

The variable *For_<N_i>* enables transfer of old values of attributes belonging to the attribute set *X* for all tuples that would be deleted. The variable *Count_IRI* is aimed to keep the number of tuples that would be deleted. Both of them are declared in the package presented in Fig. 18 and represent auxiliary data structures.

The second trigger is run just before the tuple deletion. It puts the attribute values from the tuple that would be deleted into the previously declared auxiliary data structures. The pattern for the second trigger is presented in Fig. 23.

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_DEL2
BEFORE DELETE ON <Ni>
FOR EACH ROW
DECLARE
  u <Ni>%ROWTYPE;
BEGIN
  <Initialization _u>
  <Name_P>.Count_IRI := <Name_P>.Count_IRI + 1;
  <Name_P>.For_<Ni> (<Name_P>.Count_IRI).
                    <Attr_From_X> := u.<Attr_From_X>;
                  .
                  .
                  .
END;
```

Fig. 23. A pattern of the second delete trigger

Parameter <*Name_P*> is replaced by <*Const_Name*>*_PCK*. Parameter <*Initialization _u*> is replaced by list of value assignment statements (one for each attribute from *X*), separated with the semicolons, of form:

u.<*Name_of_Attribute_from_X*> := OLD.<*Name_of_Attribute_from_X*>.

Bolded statements are repeating for each attribute from *X*.

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_DEL3
AFTER DELETE ON <Ni>
DECLARE
  u <Ni>%ROWTYPE;
  I NUMBER;
BEGIN
  FOR j IN 1.. <Const_Name>_PCK.Count_IRI LOOP
    <Initialization_u>
    SELECT COUNT(*) INTO I FROM <Nj>
    WHERE (<Selection_Cond>);
    IF I <> 0 THEN
      <Execute_Activity>
    END IF;
  END LOOP;
END;
```

Fig. 24. A pattern of the third delete trigger

The third trigger (Fig. 24) is run on the statement level after the tuple deletion. It uses the auxiliary data structures generated by the second trigger.

The replacement of parameters, specific for Oracle patterns, during the trigger generation is done as follows:

- <*Initialization_u*> is replaced by the list of value assignment statements (one for each attribute from *X*), separated with the semicolons, of form:

  u.<*Name_of_Attribute_from_X*> :=

  <*Const_Name*>*_PCK.For_*<N_i> (*j*).<*Name_of_Attribute_from_X*>;

- <*Selection_Cond*> is replaced by the conjunction of relational expressions of the form:

  <*Name_of_Attribute_From_Y*> = *u*.<*Name_of_Attribute_From_X*>.

Depending on the selected activity, <*Execute_Activity*> is replaced with *Cascade_IRI_Del_*<N_i>(*u*) procedure call (Cascade delete), that belongs to the global package *Global_PCK*, or with SQL code for activity. SQL code raises the error:

RAISE_APPLICATION_ERROR (-20003,'Tuple deletion is forbidden <N_i>').

Parameterized pattern of the procedure for cascade deletion for Oracle Server is presented in Fig. 25.

```
PROCEDURE CascadeIRI_Del_<Ni> (u IN <Ni>%ROWTYPE)
IS
BEGIN
   DELETE FROM <Nj> v WHERE (<Selection_Cond>);
END;
```

Fig. 25. A parameterized pattern of the procedure for cascade deletion

Parameter <*Selection_Cond*> is replaced by the conjunction of relational expressions of the form:

v.<*Name_of_Attribute_From_Y*> = *u*.<*Name_of_Attribute_From_X*>.

5.2.3. Patterns for tuple modification in the presence of an IRIC

Basic differences between the patterns for tuple modification in the presence of an IRIC for MS SQL Server and Oracle, are the same as the differences discussed in Section 5.2.2. Therefore, the discussion is omitted here.

Same as for tuple deletion, for Oracle 10g three triggers are to be created for the implementation of tuple modification under the presence of IRICs. The first one is run at the statement level, before the tuple modification. It has an assignment to set the auxiliary data structures, used by two other triggers. The pattern for first trigger is shown in Fig. 26.

The second trigger (Fig. 27) is run just before tuple modification. It is aimed at putting the values of attributes from *X*, for the tuple that would be modified, into the previously declared auxiliary data structures.

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_UPD1
BEFORE UPDATE ON <Ni>
BEGIN
   <Const_Name>_PCK.Count_IRI := 0;
   <Const_Name>_PCK.For_<Ni>.DELETE;
END;
```

Fig. 26. A pattern for the first modification trigger

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_UPD2
BEFORE UPDATE ON <Ni>
FOR EACH ROW
WHEN (<Cond>)
DECLARE
  u <Ni>%ROWTYPE;
BEGIN
  <Initialization_u>
  <Name_P>.Count_IRI := <Name_P>.Count_IRI + 1;
  <Name_P>.For_<Ni> (<Name_P>.Count_IRI). <Attribute_From_X> :=
        u.< Attribute_From _X>;
  .
  .
  .
END;
```

Fig. 27. A pattern for the second modification trigger

```
CREATE OR REPLACE TRIGGER TRG_<Ni>_<Const_Name>_UPD3
AFTER UPDATE ON <Ni>
DECLARE
  u <Ni>%ROWTYPE;
  I NUMBER;
BEGIN
  FOR j IN 1..<Const_Name>_PCK.Count_IRI LOOP
    <Initialization_u>;
    SELECT COUNT(*) INTO I FROM <Nj> WHERE (<Selection_Cond>);
    IF I <> 0 THEN
      RAISE_APPLICATION_ERROR
        (-20002,'Tuple modification is forbidden <Ni>');
    END IF;
  END LOOP;
END;
```

Fig. 28. A pattern for the third modification trigger

Unlike the second deletion trigger, the second modification trigger has one more parameter. That is parameter <*Cond*>. During the trigger generation process it would be replaced by the disjunction of relational expressions (one for each attribute from *X*) of the form:

NEW.<*Name_of_Attribute_from_X*> <> OLD.<*Name_of_Attribute_from_X*>.

The third trigger (Fig. 28) is run on the statement level after the tuple modification. It uses the auxiliary data structures generated by the second trigger. The replacement of the parameters is analogous to the replacement of the corresponding parameters in the third deletion trigger (Fig. 24).

6. An Example of IRIC Specification and Implementation in IIS*Studio DE

In this section, we present an example of an IRIC design specifications and transformation of design specifications into error free SQL specifications of relational DB schemas. We implement Examples 1 and 2 by means of IIS*Studio development environment.

In this section we present the processes of:

- A conceptual modelling of a DB schema;
- An automated design of relational DB schema in the 3rd normal form (3NF); and
- an automated generation of SQL/DDL code for chosen DBMSs.

A form type is the main modelling concept in IIS*Studio. Each form type is an abstraction of business documents, and therefore screens or report forms utilized by the end-users of the IS. IIS*Studio uses the set of form types to specify conceptual data model. From the set of form types, it generates the relational database schema ([19], [25]). In this way, by creating form types, a designer specifies: (i) a future database schema, (ii) functional properties of future transaction programs, (iii) and a look of the end-user interface, all at

the same time. The detailed description of the structure and specification of a form type may be found in [19] and [25]. In Fig. 29 one of IIS*Studio forms for creating form type specifications is presented.

A form type is a hierarchical structure of form type components (Fig. 29). Each component type is identified by its name within the scope of a form type, and has non-empty sets of attributes and keys, a possibly empty set of unique constraints, and a specification of the check constraint. In Example 1 a faculty is composed of at least one department. Therefore, form type *Faculty* has at least one component type *Department*. This means that each *Faculty* instance is connected with at least one *Department* instance. In Fig. 30 the IIS*Studio form for specifying component type *Department* is given. The number of occurrences of component type *Department* on the form type *Faculty* may be specified. A designer has to choose between the options: 0-N and 1-N. If the option 0-N is chosen, the model describes a faculty organization that allows the existence of a faculty with no departments. The selection of the option 1-N, models a faculty organization that does not allow the existence of a faculty with no departments. Starting from the set of form types of an IS, IIS*Studio automatically generates a relational DB schema in 3NF with all relevant data constraints.

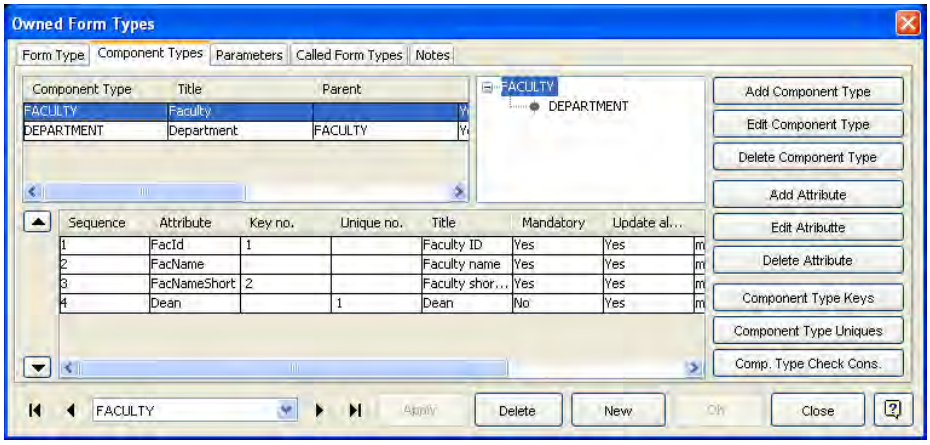


Fig. 29. The IIS*Studio form for specification of form type *Faculty*

Through the process of DB schema generation, the fact that component type *Department* is subordinated to the form type *Faculty* is recognized as the RIC *Department*[*FacId*] $\subseteq$ *Faculty*[*FacId*]. Furthermore, the selection of option 1-N for the number of occurrences of component type *Department* within the form type *Faculty* (Fig. 30) is recognized as the IRIC *Faculty*[*FacId*] $\subseteq$ *Department*[*FacId*] in the relational DB schema.

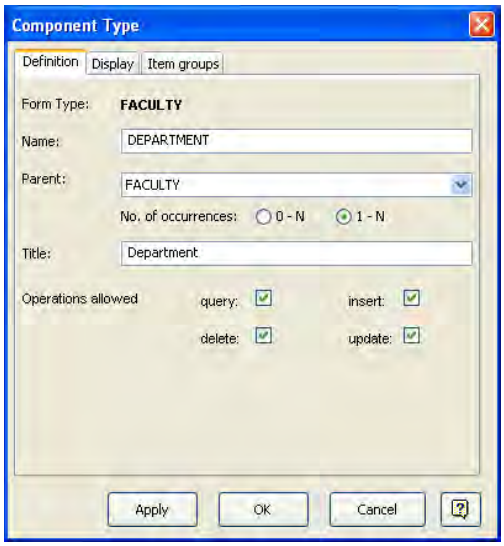


Fig. 30. The form with a specification of the component type *Department*

IIS*Studio generates a directed graph called *Closure Graph*. A graph node represents a relation scheme and a graph directed edge (arc) between two relation schemes represents an IND between them. A closure graph diagram of University database schema (UDBS) is presented in Fig. 31. The relation schemes of UDBS are represented as rectangles and INDs between them as arrows. The arrow from the *Department* to the *Faculty* rectangle represents referential integrity constraint *IND1*, while the arrow from the *Faculty* to the *Department* rectangle represents inverse referential integrity constraint *IND2*.

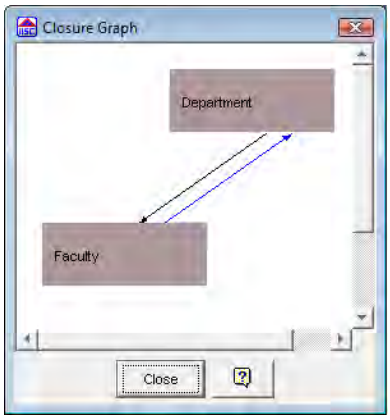


Fig. 31. A closure graph diagram of the University database schema

A designer may select an arrow representing an IND (RIC or IRIC) and invoke the appropriate form for further specifying of INDs (Fig. 32). Through

this form possible actions for keeping the DB consistency on insert, update or delete operations are to be specified. A designer may select between No Action or Cascade actions in case of an IRIC specification. This selection will affect on the corresponding deletion or modification trigger, through the way of the replacement of <*Execute_Activity*> parameter.

An automated generation of SQL/DDL code for the chosen DBMS is the next step of DB generation by using IIS*Studio. An implementation (SQL) specification of relational DB schema is generated.

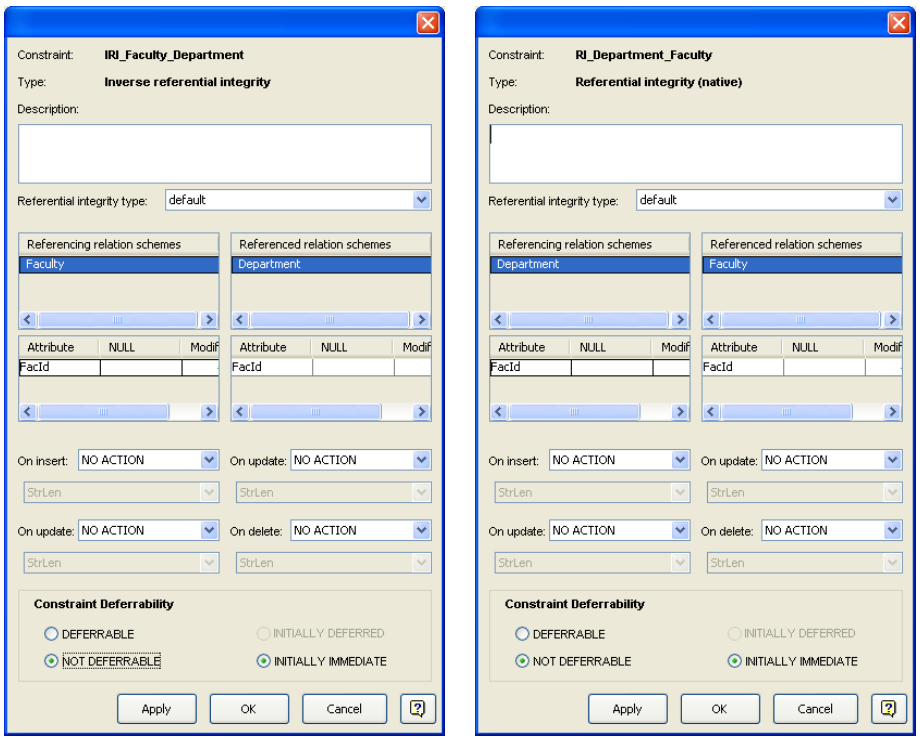


Fig. 32. Forms for specifying IRIC and RIC, respectively

Two forms that are used to define values of SQL Generator input parameters are presented in Fig. 33. Firstly, we describe the form on the left-hand side of Fig. 33. The field *DBMS* enables the selection of the type and version of a target DB server. Oracle DBMS is chosen for the example. The radio button *DDL Files only* provides the creation of SQL scripts in files only. The radio button *Database Source* enables the selection of either Oracle or MS SQL DB server, establishing a connection, and immediate execution of generated SQL scripts. In this case, SQL Generator creates a script file, invokes the appropriate SQL tool, and passes necessary parameter values for the script execution. The radio button *ODBC Source* enables the creation and the immediate execution of SQL scripts in a selected ODBC data source. An appropriate ODBC driver for the target DB server must be installed and

configured. SQL Generator supports the user authentication when it works via an established connection. The field *DB Schema Name* enables defining a DB name that is then included in an appropriate CREATE DATABASE command.

By means of *Selection* panel, a user picks relation schemes. SQL Generator will produce the appropriate SQL commands for the selected relation schemes only, and place them in script files.

By means of *Options* panel (right-hand side of Fig. 33) a user defines which types of DB objects are to be generated. By checking the appropriate check-box items, he or she may decide to generate: (i) indexes for primary, alternate and foreign keys, (ii) SQL CONSTRAINT clauses, (iii) triggers, and (iv) comments.

For inverse referential integrity constrains SQL Generator offers two ways of implementation: (i) by means of SQL views and the appropriate stored procedures, or (ii) by means of stored procedures only.

Not all possible combinations of the selected generator options are always valid. By pressing the *Check* button, a user initiates a check of the selected options. If some inconsistencies arise, a user gets the appropriate warnings. Pressing the button *Generate* initiates the generation of SQL scripts. Respecting the selected options, that can be seen on Fig. 33, the appropriate triggers are generated. The generated insertion trigger in the presence of IRIC *Faculty*[*FacId*] $\subseteq$ *Department*[*FacId*] for Oracle 10g is presented in Fig. 34. The examples of other generated procedures/triggers may be found in [1].

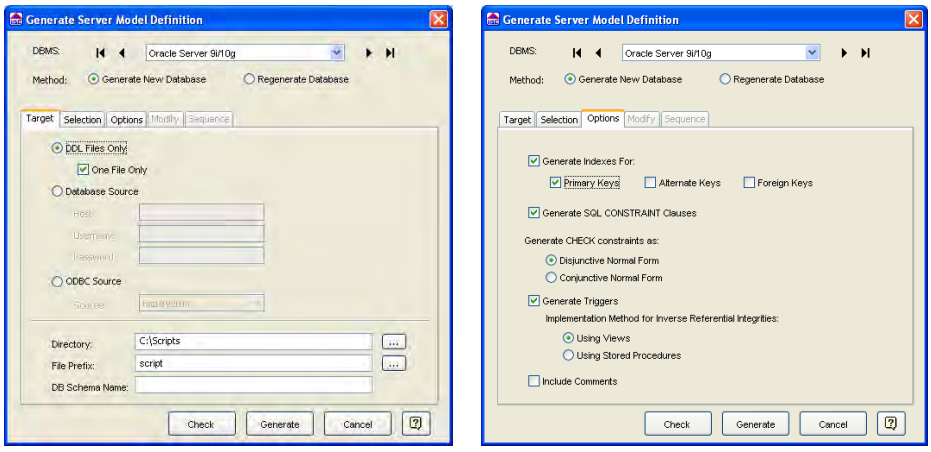


Fig. 33. A form for SQL Generator parameters specification

```
CREATE OR REPLACE TRIGGER TRG_IRI_Faculty_Department_View
INSTEAD OF INSERT ON View_Faculty_Department
FOR EACH ROW
DECLARE
   I NUMBER;
   Exc EXCEPTION;
   t Faculty%ROWTYPE;
BEGIN
   SELECT COUNT(*) INTO I
   FROM Faculty WHERE (FacId = :NEW.FacId);
   IF I <> 0 THEN
      INSERT INTO Department (FacId, DepId, DepName)
      VALUES (:NEW.FacId, :NEW.DepId, :NEW.DepName);
   ELSE
      IRI_Faculty_Department_PCK.Trigger_Ex := FALSE;
      INSERT INTO Faculty (FacId, FacName, Dean, FacShortName)
      VALUES (:NEW.FacId, :NEW.FacName, :NEW.Dean, :NEW.FacShortName);
      INSERT INTO Department (FacId, DepId, DepName)
      VALUES (:NEW.FacId, :NEW.DepId, :NEW.DepName);
      IRI_Faculty_Department_PCK.Trigger_Ex := TRUE;
      SELECT * INTO t
      FROM Faculty WHERE (FacId = :NEW.FacId);
      IF NOT Global_PCK.ContainmentIRI_Faculty(t) THEN
         RAISE Exc;
      END IF;
   END IF;
   EXCEPTION
   WHEN Exc THEN
      RAISE_APPLICATION_ERROR(-20005,'IRIC violation!');
END;
```

Fig. 34. The insertion trigger over the view for Oracle DBMS

7. Conclusion

In the paper we present an approach to the specification and implementation of IRICs. The algorithms that control the insertion, modification and deletion database operations under the presence of IRICs are shown. The patterns for triggers, as well as stored SQL functions and procedures, based on the aforementioned algorithms, are also presented. Proposed patterns provide generating SQL program code for DBMSs MS SQL Server 2008 and Oracle 10g. Our SQL Generator replaces the pattern parameters with real values obtained from a database specification stored in IIS*Case repository; then, it generates executable SQL scripts comprising necessary triggers, procedures and functions for a target DBMS platform.

While RICs are fully enforced by most current database systems, IRICs are completely disregarded by actual DBMSs, obliging users to manage them via procedural mechanisms (procedures and triggers). This implies an excessive effort to maintain integrity and develop applications. That is the reason why the IRICs are mostly implemented at the application logic (middle) layer instead at the DB server layer. Our approach enables the automated SQL scripts generation and moving of the IRICs validation at the

DB server. Thanks to that: (i) the costs of the application maintaining is cut; (ii) better performances due to the less traffic in the typical client-server architecture are provided; (iii) the same way of preventing the violation of a database consistency is enabled.

We propose two ways to insert mutually blocked tuples in one transaction: (i) a view created over the relations $r(N_i)$ and $r(N_j)$ may be used for the double insertion; or (ii) a custom DB procedure for double insertion may be developed. The first approach is more appropriate for causal end users that typically use high-level interactive query and data manipulation language. The second approach is aimed at embedding DML statements in general-purpose languages and making compiled transactions.

It is very hard to compare the features of MS T-SQL and Oracle PL/SQL in the course of implementation of IRICs. The evaluation of programmer's efforts during the program code preparing for SQL Generator strongly depends on programmer's previous knowledge, level of training and commitment to certain DBMS. We could say that according to our experience, Oracle enables easier parameters' transmission, partially due to the ROWTYPE data type, that does not exist in MS SQL Server. The ability of global variables declaration and grouping functions and procedures in packages enabled in Oracle is for sure advantage over MS SQL Server. We estimate that the existence of *Deleted* table in MS SQL Server facilitates the easier implementation of tuple deletion in the presence of IRIC, comparing with the Oracle.

Due to the fact that both Oracle and MS SQL Server, are widely used DBMSs, we decide to provide generating SQL program code for them in the first place.

Further development is directed towards extensions of SQL Generator's functionality to provide: (i) generating SQL scripts for a wider set of contemporary DBMSs and (ii) implementation of other, more complex constraints types, but often recognized in real database projects. One of typical examples is the extended referential integrity constraint (referential integrity constraint spanned over more then two relation schemes.

It is worth of emphasizing that IIS*Studio relies on the approach that conforms to the principles of model-driven (MD) approach. By means of IIS*Studio, a designer specifies only PIM models, because they are free of any implementation details. By the chain of consecutive transformations a set of different semi or fully platform specific models (PSMs) is generated. Consequently, a relational database schema that is generated by means of IIS*Studio is just one of the PSMs that we can get from PIM of the real system. Concerning the chain of model transformations, we recognize two main directions of our further research. The first one is to develop new transformations that will generate different PSMs like object-oriented model or XML model of database schemas. The second one is to develop reverse transformations from fully specific PSMs, through a series of semi PSMs towards a PIM model. One of them may be a transformation of legacy relational databases (fully PSMs) into logical database schemas expressed by the concepts of the relational data model (semi PSMs), or another, a

transformation of relational database schemas into conceptual database schemas based on the form types (PIMs). Besides, we plan to investigate a possible usage of category theory in order to improve the performance of generated code [29].

Acknowledgements. Research presented in this paper was supported by Ministry of Science and Technological Development of Republic of Serbia, Grant III-44010, Title: *Intelligent Systems for Software Product Development and Business Support based on Models*.

References

1. Aleksić S.: An SQL Generator of Database Schema Implementation Specification in a CASE Toll IIS*Case. M. Eng. (Mr.) thesis, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia. (2006)
2. Aleksić S., Luković I., Mogin P., and Govedarica M.: A Generator of SQL Schema Specifications. Computer Science and Information Systems (ComSIS), Consortium of Faculties of Serbia and Montenegro, Belgrade, Serbia, ISSN: 1820-0214, Vol. 4, No. 2, 77-96. (2007)
3. Aleksić S. and Luković I.: Generating SQL Specifications of a Database Schema for Different DBMSs. Info M - Journal of Information Technology and Multimedia Systems, Faculty of Organizational Sciences, Belgrade, Serbia, ISSN: 1451-4397, No. 23, 36-43. (2007)
4. Aleksić S., Ristić S., Luković I.: An Approach to Generating Server Implementation of the Inverse Referential Integrity Constraints. The 5th International Conference on Information Technologies ICIT 2011, May 11th – 13th, Amman, Jordan, Proceedings on CD. (2011)
5. Al-Jumaily, H.T., Cuadra, D., Martinez, P.: Plugging Active Mechanisms to Control Dynamic Aspects Derived from the Multiplicity Constraint in UML. In: The workshop of 7th International Conference on the Unified Modelling Language, Portugal. (2004)
6. Al-Jumaily H. T., Cuadra D., Martínez P.: OCL2Trigger: Deriving active mechanisms for relational databases using Model-Driven Architecture. Journal of Systems and Software, Vol. 81, No. 12, 2299-2314, ISSN 0164-1212, (2008)
7. ARTech. DeKlarit™ (The Model-Driven Tool for Microsoft Visual Studio 2005), Chicago, U.S.A. [Online]. Available: http://www.deklarit.com/. (2007)
8. ANSI SQL:2003, American National Standards Institute, USA, ISO/IEC Std. 9075-{1, 2, 11}. (2003)
9. Ataullah A. A., Tompa F.: Business Policy Modelling and Enforcement in Databases. PVLDB Vol. 4, No. 11, 921-931. (2011)
10. Badaway, M., Richta, K.: Deriving triggers from UML/OCL specification, Information Systems Development: Advances in Methodologies, Components and Management, 305-315. (2003)
11. Berrabah D., Boufarès F.: Constraints Checking in UML Class Diagrams: SQL vs OCL. Database and Expert Systems Applications Lecture Notes in Computer Science, Vol. 4653/2007, 593-602, DOI: 10.1007/978-3-540-74469-6_58. (2007)
12. Bravo, L.: Handling Inconsistency in Databases and Data Integration Systems. Ph.D. Thesis, Carleton University. (2007)

13. Cabot J., Teniente E.: Constraint Support in MDA tools: A Survey. Proceedings of 2nd European Conference on Model Driven Architecture, LNCS, ECMDA-FA, 256-267. (2006)
14. CA ERwin Data Modeler r7.3. [Online]. Available: https://support.ca. com/irj/. (2008)
15. Ceri, S., Cochrane, R., and Widom, J.: Practical applications of triggers and constraints: success and lingering Issues. In VLDB 2000, 254-262, (2000)
16. Decker H., Martinenghi D.: Database Integrity Checking. In Database Technologies: Concepts, Methodologies, Tools, and Applications, ed. John Erickson, 212-220, doi:10.4018/978-1-60566-058-5.ch016. (2009)
17. Elmasri R., Navathe B. S.: Database Systems: Models, Languages, Design and Application Programming, Sixth Edition, Pearson Global Edition, ISBN 978-0-13-214498-8. (2011)
18. Govedarica M.: Design the Set of Implementation Database Schema Constraints. M. Eng. (Mr.) thesis, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia. (1998)
19. Luković I., Mogin P., Pavićević J., and Ristić S.: An Approach to Developing Complex Database Schemas Using Form Types. Software: Practice and Experience, John Wiley & Sons Inc, Hoboken, USA, ISSN: 0038-0644, DOI: 10.1002/spe.820 Vol. 37, No. 15, 1621-1656. (2007)
20. Luković I., Ristić S., Mogin P., and Pavicević J.: Database Schema Integration Process – A Methodology and Aspects of Its Applying, Novi Sad Journal of Mathematics, Faculty of Science, Novi Sad, Serbia, ISSN: 1450-5444, Vol. 36, No. 1, 115-140. (2006)
21. Microsoft SQL Server 2008. (2008)
22. Mogin P., Luković I., and Govedarica M.: Database Design Principles, 2nd Edition, University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia, ISBN: 86-80249-81-5. (2004)
23. Mogin P., Luković I., and Govedarica M.: Extended Referential Integrity, Novi Sad Journal of Mathematics, Novi Sad, Serbia, ISSN: 1450-5444, Vol. 30, No. 3, 111-122. (2000)
24. Oracle DBMS 10g. (2004)
25. Pavićević J., Luković I., Mogin P., and Govedarica M.: Information System Design and Prototyping Using Form Types. International Conference on Software and Data Technologies, Setubal, Portugal, September 11-14, Vol. 2, 157-160. (2006)
26. Ristic S., Aleksic S., Lukovic I., Banovic J.: Form-Driven Application Generation: A Case Study, In Proceedings of the XI International Conference on Informatics, Roznava, Slovakia, 115 – 120. (2011)
27. Rybola Y., Richta K.: Transformation of binary relationship with particular multiplicity. In DATESO 2011, Vol. 11, 25–38. Czech Republic: Department of Computer Science, FEECS VSB – Technical University of Ostrava. [Online]. Available: http://www.informatik.uni-trier.de/ ˜ley/db/conf/dateso/dateso 2011.html. (2011)
28. Sybase PowerDesigner 15. (2009)
29. Slodičák V.: Some useful structures for categorical approach for program behaviour. Journal of Information and Organizational Sciences, Vol. 35, No. 1, 99-109, (2011)
30. Türker, C., Gertz, M.: Semantic Integrity Support in SQL-99 and Commercial (Object) Relational Database Management Systems. UC Davis Computer Science Technical Report CSE-2000-11, University of California. (2000)

31. Zimbrão, G., Miranda, R., de Souza, J., Estolano, M.H, Neto, F. P.: Enforcement of business rules in relational databases using constraints. In Proceedings of XVIII Simposio Brasileiro de Bancos de Dados/SBBD, 129-141, UFAM. (2003)

Slavica Aleksić received her M.Sc. degree from the Faculty of Technical Sciences at University of Novi Sad. She completed her Mr (2 year) degree at the University of Novi Sad, Faculty of Technical Sciences. Currently, she works as a teaching assistant at the Faculty of Technical Sciences, at University of Novi Sad, where she assists in teaching several Computer Science and Informatics courses. Her research interests are related to Database Systems, Theory of Data Models, System Design, Logical and Physical Database Design, Development and Usage of MDSE / CASE tools in Software Engineering and System Design, Reengineering of Information Systems and Model Transformations in MDA.

Sonja Ristić works as an associate professor at the University of Novi Sad, Faculty of Technical Sciences, Serbia. She received two bachelor degrees with honors from University of Novi Sad, one in Mathematics, Faculty of Science in 1983, and the other in Economics from Faculty of Economics, in 1989. She received her Mr (2 year) and Ph.D. degrees in Informatics, both from Faculty of Economics, University of Novi Sad, in 1994 and 2003. From 1984 till 1990 she worked with the Novi Sad Cable Company NOVKABEL– Factory of Electronic Computers. From 1990 till 2006 she was with High School of Business Studies -Novi Sad, and since 2006 she has been with the Faculty of Technical Sciences at University of Novi Sad. Her research interests are related to Database Systems and Software Engineering.

Ivan Luković received his M.Sc. degree in Informatics from the Faculty of Military and Technical Sciences in Zagreb in 1990. He completed his Mr (2 year) degree at the University of Belgrade, Faculty of Electrical Engineering in 1993, and his Ph.D. at the University of Novi Sad, Faculty of Technical Sciences in 1996. Currently, he works as a Full Professor at the Faculty of Technical Sciences at the University of Novi Sad, where he lectures in several Computer Science and Informatics courses. His research interests are related to Database Systems and Software Engineering. He is the author or co-author of over 90 papers, 4 books, and 30 industry projects and software solutions in the area.

Milan Čeliković graduated in 2009 at the Faculty of Technical Sciences, Novi Sad, at the Department of Computing and Control. Since 2009 he has worked as a teaching assistant at the Faculty of Technical Sciences, Novi Sad, at the Chair for Applied Computer Science. In 2010, he started his Ph.D. studies at the Faculty of Technical Sciences, Novi Sad. His main research interests are focused on: Domain specific modeling, Domain specific languages, Databases and Database management systems. At the moment, he is involved in the projects concerning application of DSLs in the field of software engineering.

Received: November 02, 2011; Accepted: December 08, 2012.

DOI:10.2298/CSIS120601006P

Methods for Division of Road Traffic Network for Distributed Simulation Performed on Heterogeneous Clusters

Tomas Potuzak[1]

[1] University of West Bohemia, Department of Computer Science and Engineering,
Univerzitni 8, 306 14 Plzen, Czech Republic
tpotuzak@kiv.zcu.cz

Abstract. The computer simulation of road traffic is an important tool for control and analysis of road traffic networks. Due to their requirements for computation time (especially for large road traffic networks), many simulators of the road traffic has been adapted for distributed computing environment where combined power of multiple interconnected computers (nodes) is utilized. In this case, the road traffic network is divided into required number of sub-networks, whose simulation is then performed on particular nodes of the distributed computer. The distributed computer can be a homogenous (with nodes of the same computational power) or a heterogeneous cluster (with nodes of various powers). In this paper, we present two methods for road traffic network division for heterogeneous clusters. These methods consider the different computational powers of the particular nodes determined using a benchmark during the road traffic network division.

Keywords: road traffic simulation, network division, distributed simulation, heterogeneous clusters.

1. Introduction

The computer simulation of road traffic is an important tool for control and analysis of existing or designed road traffic networks. However, a run of a detailed simulation of a large road traffic network (e.g. entire cities or even states) can be very time-consuming. Moreover, very often, multiple simulation runs are required in order to ensure fidelity of the collected results. Hence, many simulators of the road traffic have been adapted for the distributed computing environment. There, the combined power of multiple interconnected computers (nodes) is utilized for speedup of the simulation.

The adaptation for the distributed computing environment usually means that the road traffic network is divided into required number of sub-networks. Simulations of these sub-networks are then performed as processes on particular nodes of the distributed computer. In order to fully exploit the power

of each node and therefore maximize the speed of the distributed simulation, the particular sub-networks should be load-balanced.

If the computer, on which the simulation is running, is a homogenous cluster (i.e. with nodes of the same computational power), the load of the particular sub-networks should be similar. Nevertheless, quite often, the nodes of the distributed computer can be of different computational power (e.g. various desktop computers interconnected by Ethernet in a university campus). For such a heterogeneous cluster, the load-balancing means that the load of each sub-network is adapted for the computational power of the node, on which the simulation of this sub-network will be performed.

In the reminder of this paper, two methods for road traffic network division for heterogeneous clusters are described. These methods consider the different computational powers of the particular nodes during the division. The information about the actual power of each node is determined using a set of tests (i.e. a benchmark) rather than using the information about the processor speed, memory size, and so on. The methods have been thoroughly tested and compared each other and with methods for homogenous clusters. The results of the testing are part of this paper as well.

2. Distributed Road Traffic Simulation

The methods for division of road traffic networks presented in this paper are designed for distributed discrete time-stepped microscopic simulation of road traffic. Moreover, the methods themselves utilize less-detailed road traffic simulations. Hence, the basic features and issues of the road traffic simulation and its distributed version are briefly described in following sections.

2.1. Time-Flow Mechanism of the Simulation

One of the most important features of a general simulation is the way the simulation time is advanced (i.e. a time-flow mechanism). There are two commonly used approaches – the *time-stepped* time flow mechanism and the *event-based* time-flow mechanism [1].

Using the time-stepped time-flow mechanism, the entire simulation time is subdivided into sequence of equally-sized time steps. The length of the time step is often set to one second. In each time step, the entire simulation state is recomputed [1].

Using the event-driven mechanism, the simulation time is subdivided into sequence of events. With each event, an action and a time stamp are associated. The action is an incremental change of the simulation state and the time stamp determines when this change shall happen [1].

2.2. Level of Detail of the Road Traffic Simulation

There are several types of road traffic simulation, which can be most commonly divided based on the level of detail into *macroscopic*, *mesoscopic*, and *microscopic* simulations.

In a macroscopic simulation, the traffic in particular traffic lanes is represented by traffic flows described by set of parameters (e.g. mean speed, vehicle density, etc.). These parameters are periodically recomputed. The macroscopic simulations are the oldest ones [2] and exists in many modifications. Both mentioned time-flow mechanisms (see Section 2.1) are commonly used. Since there are no vehicles considered in the simulation and the traffic flows are represented only by a limited number of parameters, the macroscopic simulation can be very fast. It is often possible to simulate a very large road traffic network faster than in a real time on a standard desktop computer [3].

The microscopic simulation represents the opposite side of the road traffic simulations field. In this simulation type, every single vehicle is considered with its own position, direction, speed, and acceleration. The positions of the vehicles are periodically recomputed based on their current speed and the utilized traffic model. Two basic models are commonly used – the *cellular automaton model* [4] and the *car-following model* [5]. Using the cellular automaton model, the traffic lanes are divided into equally-sized cells. The vehicles can be placed only into these cells and are moving from a cell to another based on their current speed. Using the car-following model, the vehicles can be placed anywhere in the traffic lane. The first vehicle in the lane can move freely, but the other vehicles must respect the speed of the vehicles in front of them. In both models, the vehicles tend to accelerate to a maximal speed, if there are no obstacles (e.g. a slower vehicle) in their way, and slow down or even stop otherwise [4], [5]. A random slowdown is often used as a component of the speed of the vehicles in order to model its natural fluctuations due to the traffic conditions [4]. Regardless the utilized traffic model, in the vast majority of the microscopic road traffic simulations, the time-stepped time-flow mechanism is used (for example, see [6] and [7]). Due to the high detail of the simulation, the collected results are more precise, but the simulation runs are very time-consuming, especially for large traffic networks. The majority of the computation time is usually consumed by the movement of the vehicles [8].

The mesoscopic simulation lies between the macroscopic and the microscopic simulations described in previous paragraphs. There are many various approaches, which are considered mesoscopic, such as queuing networks [9] or gas-kinetic models [10]. However, the common characteristic is that there is a representation of the vehicles, but their interactions are modeled at very low detail [11]. So, a mesoscopic simulation can be much faster than a microscopic simulation. Both time-flow mechanisms are commonly used.

2.3. Decomposition of the Road Traffic Simulation

When a road traffic simulation is prepared for the distributed computing environment, it must be decomposed in some way in order to be performed on more than one computer (or node). There are several general types of decomposition of a simulation. However, in the field of road traffic simulation, the *spatial decomposition* is most common. Using this approach, the road traffic network is divided into a required number of sub-networks (i.e. parts of the original road traffic network). Simulation of each sub-network is then performed as a process on a node of the distributed computer.

The number of sub-networks often corresponds to the number of nodes of the distributed computer (to maximize efficiency). Nevertheless, it is possible to perform more simulation processes on one node [7]. If the node has multiple processors or multiple processor cores, this can lead to further speedup of the simulation and better exploitation of the node's computational power [12]. In this paper, we will not consider this possibility further, unless otherwise stated.

There are other approaches to the simulation decomposition, such as *task parallelization* or *temporal decomposition*. Since their utilization in the field of road traffic simulation is very rare (some examples can be found in [13] and [14], respectively), we will not consider them further.

2.4. Inter-process Communication in Distributed Traffic Simulation

Using the spatial decomposition, the road traffic network is divided into required number of sub-networks, which are then simulated by simulation processes running on the particular nodes of the distributed computer. These sub-networks were originally interconnected by a set of traffic lanes, which are divided during the decomposition. However, the interconnection of the sub-networks must be maintained in the distributed simulation in order to enable passing of the vehicles in the divided lanes. For this purpose, communication links are established among the simulation processes. The vehicles passing from one sub-network to another are then transferred as messages between the corresponding simulation processes [3].

Besides the transfer of vehicles, it is also necessary to ensure the consistency of the entire distributed simulation. This means that all vehicles passing among the sub-networks must arrive to the target sub-network in correct simulation time. Otherwise, a causality error occurs [1]. Considering the time-stepped time-flow mechanism only (see Section 2.1), this means that a vehicle arrives in an incorrect time step (i.e. past or future). So, all simulation processes must perform the same time step at the same moment to maintain the consistency of the distributed simulation. This is ensured by a synchronization mechanism based on a synchronization barrier. This mechanism allows the simulation process to continue with next time step after all simulation processes finished the computation of the previous time

step. The barrier can be implemented in a separate process running on another node of the distributed computer or can be distributed [3].

Both the transfer of vehicles and the synchronization are maintained by a communication protocol of the distributed road traffic simulation. Because the inter-process communication is relatively slow in comparison to the remainder of the computations of the distributed simulation, it is convenient to reduce it. This can be achieved in several ways, for example by aggregation of more vehicles into one message spatially (vehicles from more traffic lanes) or temporally (vehicles from more time steps). A detailed discussion of the possibilities of the reduction of inter-process communication is outside the scope of this paper, but further information can be found in [3]. Nevertheless, the communication can be positively influenced even by a convenient division of the traffic network when the number of divided traffic lanes is minimized (see Section 3).

2.5. Distributed Road Traffic Simulator for Testing

The methods for division of road traffic networks described later in the text have been tested using the Distributed Urban Traffic Simulator (DUTS), which has been developed at Department of Computer Science and Engineering of University of West Bohemia (DSCE UWB). It is a distributed discrete time-stepped simulator of urban road traffic, but can be performed on a single-processor computer as well [15].

The simulator incorporates three traffic models inspired by three existing road traffic simulators – one car-following model (inspired by the AIMSUN [16] simulator) and two cellular automaton models (inspired by the JUTS [17] and TRANSIMS [6] simulators, respectively). Because the methods for the division of road traffic networks are independent on the traffic model utilized for the simulation, only the JUTS-based traffic model was used for testing. The reason is that this model has medium computational demands from all three models [3].

The DUTS simulator also incorporates several communication protocols of different efficiency. For the testing, a basic SC-LV (Semi-Centralized Lane Vehicles) rather than an advanced communication protocol was used. The SC-LV protocol transfers all vehicles from one traffic lane in one time step in one message. Usually, each message contains only one vehicle. However, if there are two or more vehicles traveling from a sub-network to a neighboring one in one traffic lane in one time step, they are transferred together in one message [3]. The synchronization is performed in every time step using a control process with a centralized barrier. Each working process simulating a traffic sub-network exchanges two messages with the control process per time step [3]. The SC-LV protocol was used because similar protocols are used in many existing distributed road traffic simulators. More importantly, it enables better testing of the quality of the methods for division of road traffic networks regarding the number of divided traffic lanes. A more advanced protocol could mitigate the influence of the number of divided traffic lanes [3].

3. Common Approaches to Road Traffic Network Division

Now, as we discussed the general features and issues of the distributed road traffic simulation, we can focus on the road traffic network division. There are two main issues, which should be considered during the road traffic network division – the load-balancing of the resulting sub-networks and the inter-process communication. Both issues are important for the resulting performance of the distributed road traffic simulation, for which the road traffic network is divided.

The load-balancing is important because of the synchronization. All simulation processes perform the same time step at the same moment (see Section 2.4). This means that the faster processes must wait until the slower processes finish the computation in each time step [3]. Hence, it is desirable for all simulation processes to consume similar amount of time for the computation of each time step [18]. If the distributed computer is a cluster of homogenous computers, the road traffic network should be divided into sub-networks with similar load (i.e. similar numbers of vehicles moving within the sub-networks).

The inter-process communication is important, because it is very slow in comparison to the remainder of the computations of the distributed simulation. Hence, it is desirable to reduce the inter-process communication to the necessary minimum. Because the communication is necessary primarily for the transfer of vehicles among the sub-networks, its intensity depends on the number of traffic lanes divided during the road traffic network division and also on the vehicle densities in these lanes [3].

There are several existing approaches to the road traffic network division, which can but do not have to consider the mentioned issues. Some of them are described in following sections.

3.1. Division without Any Optimization

The easiest approach is to divide the road traffic network into equally-sized rectangular pieces (sub-networks). Using this division, neither the load-balancing nor the inter-process communication is optimized.

Utilization of this approach can be found for example in *ParamGrid* simulator [19]. The main disadvantage of this approach is that the density of the traffic lanes in the particular rectangular pieces and the vehicle density within these lanes are not considered during the road traffic network division. These two parameters of the road traffic network can seriously affect the number of vehicles moving in the particular rectangular pieces (sub-networks) during the simulation run. Hence, the loads of the resulting sub-networks can be very different. Moreover, the number of divided traffic lanes is not considered during the road traffic network division. This can lead to a high number of divided traffic lanes and therefore to an intensive inter-process communication [3].

3.2. Optimization of the Inter-process Communication

A more advanced approach can be found in the TRANSIMS simulator [6]. The road traffic networks division used in this simulator is focused on minimization of the number of divided traffic lanes and the number of sub-networks' neighbors. Graph-partitioning methods such as orthogonal recursive bisection are used for this purpose. The lower number of neighbors and divided traffic lanes leads to a reduction of the inter-process communication.

The load-balancing of the resulting road traffic sub-networks are partially considered as well, since the total length of the traffic lanes in each sub-network is considered during the division [6]. Nevertheless, the total length of the traffic lanes is not sufficient to guarantee the load-balancing of the sub-networks due to the possible various vehicle densities in particular traffic lanes.

3.3. Static Load-Balancing

The issue mentioned in previous section is solved in the UMTSS simulator [20]. Similar to the TRANSIMS simulator, a recursive bisection method is used for the road traffic network division. However, besides the length of the traffic lanes, the vehicle densities in these lanes are considered as well. The information of the vehicle densities in the traffic lanes is estimated using the drivers' route choice decision and the origin-destination matrix [20].

Another approach focused on the load-balancing of the sub-networks can be found in the *vsim* simulator [7]. In this simulator, the number of divided traffic lanes is completely neglected. The division is performed based on the numbers of vehicles moving within the lanes of the road traffic network. This information is collected during a sequential simulation run of the distributed simulation [7].

Although this approach has the potential to produce a well load-balanced sub-networks, its main issue is the collection of the numbers of vehicles using a sequential simulation run. A sequential run of a simulation intended to be performed on a distributed computer can be difficult to perform on a single-processor computer due to the time and memory requirements [15].

4. Division Methods for Heterogeneous Clusters

The methods for the division of road traffic networks for the heterogeneous clusters are based on the methods originally developed solely for the homogenous clusters. Both the methods for the homogenous and the heterogeneous clusters are implemented in the DUTS Editor, a system for the design and division of road traffic networks developed at DSCE UWB.

We have developed two methods for the homogenous clusters. Both utilize the weights representing numbers of vehicles assigned to the particular traffic lanes for the load-balanced division of the road traffic network. Also, both methods divide the road traffic network by marking of traffic lanes, which shall be divided in order to form the required number of sub-networks. The difference between the methods is the approach used for the marking of traffic lanes. The assigning of the weights to the traffic lanes is described in Section 4.1. The division methods themselves are described in Section 4.2 and Section 4.3.

4.1. Assigning of the Weights to the Traffic Lanes

The assigning of the weights to the traffic lanes is the basis for both methods for road traffic network division for homogenous clusters. We have developed three approaches to the weights assigning (WA). All three can be used by both methods and all three utilize a road traffic simulation for counting of the vehicles moving within the particular lanes of the road traffic network. The weight of each traffic lane is then calculated as the mean number of vehicles moving within the lane during the simulation run [21]. The particular approaches to the weights assigning, which are described in following paragraphs, differ mainly in the utilized simulation.

The MaSBWA (Macroscopic-Simulation-Based Weights Assigning) approach uses a deterministic macroscopic road traffic simulation. Since there are no pseudo-random numbers utilized, all simulation runs of a single traffic network are identical. Hence, only one simulation run is required for the calculation of the weights of the traffic lanes, which makes this approach very fast [22].

The MeSBWA (Mesoscopic-Simulation-Based Weights Assigning) approach uses several simulation runs of a stochastic mesoscopic road traffic simulation based on a simple cellular automaton [21]. Because the simulation is not deterministic, several simulation runs are required in order to guarantee the fidelity of the calculated weights. This makes the MeSBWA approach slower than the MaSBWA approach [21].

Nevertheless, the far slowest approach is the MiSBWA (Microscopic-Simulation-Based Weights Assigning). The reason is the direct utilization of the microscopic simulation runs of the DUTS system (similar to the *vsim* simulator – see Section 3.2). These simulation runs have quite extreme time requirements on a standard desktop computer due to the high detail [22]. As an example, we can use a single 15-minutes-long simulation run of a road traffic network with 1 024 crossroads and over 1 200 kilometers of traffic lanes performed on an average desktop computer. Using the MiSBWA, the simulation run takes approximately 2 minutes to be performed. In comparison, the MaSBWA method requires only 6 seconds [8].

Based on the performed tests, all approaches for weights assigning give comparable results [21], [22]. Hence, it is convenient to use the fastest

approach, which is the MaSBWA. So, the MaSBWA approach is used in both road traffic network division methods for homogenous clusters by default.

4.2. MBFSMTL Method for Homogenous Clusters

As has been said, we have developed two methods for road traffic network division for homogenous cluster. The first method is the MBFSMTL (Modified Breadth-First Search Marking of Traffic Lanes), which employs a modified breadth-first searching algorithm for graph exploration [23] for the creation of the load-balanced sub-networks. The number of divided traffic lanes is partially considered as well.

The MBFSMTL method considers the road traffic network as a weighted graph with crossroads acting as nodes and the sets of lanes inter-connecting the neighboring crossroads acting as edges. This graph is then explored from a staring crossroad using the breadth-first search algorithm. The crossroads are assigned to the particular sub-networks based on the actual sum of the weights of the explored edges (i.e. sets of traffic lanes). Once the entire traffic network is explored, the lanes connecting crossroads from different sub-networks are marked to be divided [15]. The exploration of the road traffic network is depicted in Fig. 1.

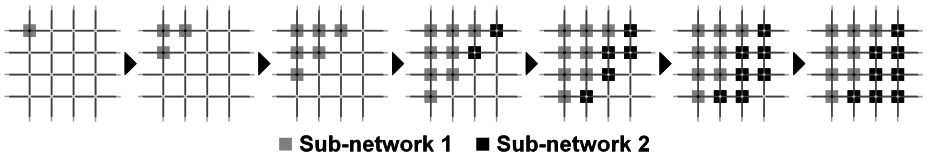


Fig. 1. The exploration of the road traffic network using MBFSMTL

Because the result of the division is significantly influenced by the selection of the starting crossroad, the entire process is performed from all crossroads of the traffic network. The division with the minimal number of divided traffic lanes is then selected [15].

4.3. GAMTL Method for Homogenous Clusters

The second method is the GAMTL (Genetic Algorithm Marking of Traffic Lanes). It employs a standard genetic algorithm for a multi-objective optimization [24] for the division of the road traffic network.

The genetic algorithms mimic the natural genetic evolution in nature. At the beginning, a set of solutions (so-called *individuals*) of the solved problem is (most often randomly) generated. This set is called *initial population*. Then, for each individual, a so-called *fitness value* is calculated using the *fitness function*. This value is an objective assessment of the individual in relation to the solved problem. The better the individual (i.e. solution of the problem) is,

the higher its fitness value is. Once this value is calculated for entire population, a number of individuals are *selected*, *crossed*, and *mutated* in order to produce a new generation. This process repeats until a preset fitness value is reached or a preset number of generations is created [25].

Using the GAMTL method, an individual is a specific assignment of the crossroads to the particular sub-networks. The fitness function represents the requirements on the solution – the load-balancing of the resulting sub-networks (called *equability*) and the minimal number of divided traffic lanes (called *compactness*). The ratio between these two parts of the fitness function can be set prior the road traffic network division similar to the number of generations and the number of mutations per individual [26]. The assignment of the crossroads to the particular sub-networks then changes using the crossover and mutation from a random pattern to clusters of crossroads (see Fig. 2).

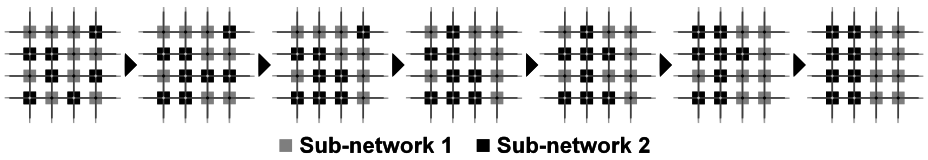


Fig. 2. The change of the assignment of the crossroads using the GAMTL

The assignment of the crossroads to the particular sub-networks represented by the individual of the last generation with the highest fitness value is used for the division of the road traffic network. Similar to the MBFSMTL method, the lanes connecting the crossroads assigned to different sub-networks are marked as divided.

Based on the performed tests, the GAMTL method gives slightly better results. This means that the distributed simulation of the road traffic network divided by the GAMTL method is slightly faster (from 3 % to 22 %) than the simulation of the same traffic network divided by the MBFSMTL method [8]. On the other hand, the computation time of the MBFSMTL method is lower then the computation time of the GAMTL method. Hence, both methods seem to be utilizable [27].

5. Division Methods for Heterogeneous Clusters

For the heterogeneous clusters, the methods originally developed for the homogenous clusters (see Section 4) must be modified.

5.1. Specifics of the Heterogeneous Cluster

The key difference of a heterogeneous cluster in comparison to a homogenous cluster is that the particular nodes have different computational

powers [28]. This makes the load-balancing of the road traffic sub-networks a little more complicated task. The division of the road traffic network into the sub-networks with similar number of vehicles (i.e. similar load) is impractical in this case [28]. Using such a division, the computational power of the slowest node would be fully utilized, but the faster nodes would have to wait for this slowest node to finish the computation in every time step.

Hence, it is necessary to take the computational power of each node of the heterogeneous cluster into the account during the road traffic network division. This means that each node is assigned by such a part of the road traffic network that the computation of one time step takes the similar time to all nodes. Then, the computational power of each node is fully utilized and the speed of the distributed simulation is maximized [27].

5.2. Necessary Modifications to the Division Methods

As mentioned in Section 5, the road traffic division methods developed for the homogenous clusters must be modified in order to be fully utilizable and efficient for the heterogeneous clusters.

Nevertheless, the assigning of the weights to traffic lanes used by both methods requires no modifications. In this computation, the data are only prepared for the marking of traffic lanes and no information about the target distributed computer it utilized. All three approaches to the weights assigning described in Section 3.1 could be used for the road traffic network division for the heterogeneous clusters. However, because all the approaches give similar results [21], [22], the fastest approach – the MaSBWA – will be used.

On the contrary, the modifications are necessary to the marking of traffic lanes of both methods. It is necessary to consider the various computational powers of the nodes of the heterogeneous cluster during the creation of the resulting sub-networks. This means that the load of the traffic network must not be divided uniformly among the resulting sub-networks, but rather in ratio, which corresponds to the ratio of the computational powers of the nodes of the cluster. However, this ratio is specific for each heterogeneous cluster and must be determined prior to the division of the road traffic network [27].

5.3. Investigation of the Speed of Heterogeneous Cluster Nodes

The determination of the computational power (i.e. speed) of the particular nodes of the heterogeneous cluster is vital for the successful division of the road traffic network. From the information about the speed of the particular nodes, the computational power ratio can be easily calculated. This ratio will be then used for the division of the load of the road traffic network among the resulting sub-networks. There are several possibilities how to determine the speed of the node [27].

The first possible approach is the utilization of known parameters of the node, such as CPU frequency, size of the RAM, number of floating-point operations per second, and so on. Nevertheless, from this information, it is difficult to determine the resulting speed of the road traffic simulation performed on the node. This information is most important for the determination of the computational power ratio of the particular nodes. Some theoretical speed of the nodes is from this point of view irrelevant [27].

Hence, it is more convenient to use a set of tests (i.e. a benchmark) for determination of the speeds of the particular nodes of the heterogeneous cluster. These tests should utilize directly the road traffic simulation in order to obtain most relevant information about the speeds of the nodes. This approach is used for both our methods for the division of road traffic network for heterogeneous clusters [27].

For the determination of the speed of each node of the heterogeneous cluster, the DUTS simulator (see Section 2.5) is used. Three road traffic networks were prepared to be performed on each node. The networks are regular grids of 16, 64, and 256 crossroads (see Fig. 3). The networks are small enough to keep the testing time in reasonable limits even on slower nodes and large enough to make the testing easily measurable [27].

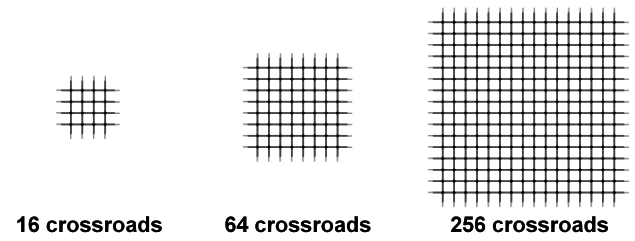


Fig. 3. The road traffic networks used for benchmarking of each node

During the benchmark, ten simulation runs of each road traffic network are performed on every node of the heterogeneous cluster. The computation time and the mean number of vehicles moving within the road traffic network are observed during each simulation run. For each road traffic network and each node, the mean computation time and the mean number of vehicles are calculated. It has been determined that ten simulation runs for each combination of the node and road traffic network are sufficient. The mean deviation of each measured value from the mean value is under 1 % for the computation time and under 3 % for the mean number of vehicles moving within the traffic network [27].

With the calculated mean computation time and mean number of vehicles for each road traffic network, the computational power coefficient for each node of the heterogeneous cluster can be determined using equation:

$$k_i = \frac{\sum_{j=1}^{3} \frac{V_{ij}}{T_{ij}}}{3}, \tag{1}$$

where k_i is the computational power coefficient of the ith node, V_{ij} is the mean number of vehicles of the jth road traffic network on the ith node, and T_{ij} is the mean computation time of the jth traffic network on the ith node. Using the coefficients, the portion of the load of each node can be calculated as:

$$r_i = \frac{k_i}{\sum_{j=1}^{N} k_j}, \tag{2}$$

where r_i is the portion of the load of the ith node, k_i is the computational power coefficient of the ith node, k_j is the computational power coefficient of the jth node, and N is the number of nodes. The calculated portions of the load of the particular nodes are then used for division of the load of the road traffic network among the sub-networks in ratio $r_1 : r_2 : \ldots : r_N$.

5.4. MBFSMTL Method for Heterogeneous Clusters

Now, as we discussed the determination of the ratio, which will be used for division of the load among the road traffic sub-networks, we can proceed with the description of the methods for marking of traffic lanes for heterogeneous clusters. Both described methods for homogeneous clusters are viable for heterogeneous clusters as well, but with some modifications. The MBFSMTL method for heterogeneous clusters is described in this section. The GAMTL method for heterogeneous clusters is described in Section 5.5.

As it was said in Section 4.2, the MBFSMTL method utilizes a modified breadth-first search algorithm. The inputs of the method are the road traffic network with traffic lanes assigned with weights (see Section 4.1 and 5.2) and the portions of the load of the particular nodes (see Section 5.3). The road traffic network is considered as a weighted graph with crossroads acting as nodes of the graph and the sets of lanes connecting crossroads acting as weighted edges of the graph (see Fig. 4). Prior to the searching of the graph, the total weight of the divided traffic network is calculated as:

$$W_T = \sum_{i=1}^{L} w_i, \tag{3}$$

where W_T is the total weight of the divided traffic network, L is the number of traffic lanes of the road traffic network, and w_i is the weight of the ith lane.

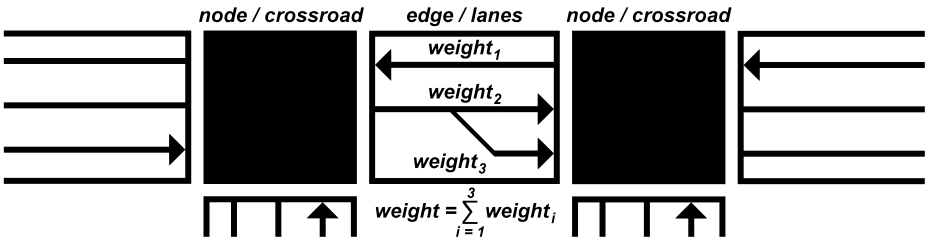


Fig. 4. Road traffic network as weighted graph (an input for division methods)

Once the total weight of the road traffic network is determined, the aimed weights of particular sub-networks can be calculated. Unlike the MBFSMTL method for homogenous clusters, this number is not the same for all sub-networks, but must be calculated using the portions of the load of the particular nodes of the heterogeneous cluster as:

$$W_{Si} = r_i \cdot W_T \,, \tag{4}$$

where W_{Si} is the aimed weight of the *i*th sub-network, r_i is the portion of the load of the node, on which the simulation of the *i*th sub-network will be performed, and W_T is the total weight of the divided traffic network. Also, the current sub-network's number is set to zero, the current sub-network's current weight is set to zero, and the aimed weight of current sub-network is set to the first value of the aimed weights of the particular sub-networks (i.e. W_{S1}).

When all necessary values are initialized, a crossroad is selected as the starting node of the breadth-first search algorithm. The starting crossroad is assigned by the current sub-network's number (zero at this point). As the breadth-first searching is performed, the explored crossroads are assigned with the current sub-network's number and the weights of the explored edges are added to the current sub-network's current weight. When this value reaches the aimed weight of the current sub-network, the current sub-network's number is incremented, the current sub-network's current weight is reset to zero, and the aimed weight of the current sub-network is set to the next value of the aimed weights of the particular sub-networks. These steps repeat until the entire road traffic network is explored [27].

When the entire road traffic network is explored, it means that all crossroads have been assigned with the number of road traffic sub-network. To mark traffic lanes, which shall be divided to create the sub-networks, it is sufficient to mark all traffic lanes connecting all pairs of the crossroads with different numbers of sub-networks [27].

Similar to the MBFSMTL method for homogenous clusters, the breadth-first search is performed from all crossroads of the divided road traffic network. The results of the division (i.e. the sets of traffic lanes marked to be divided) are stored in the memory. When the divisions for all possible starting crossroads are finished, the division with minimal number of divided traffic lanes is selected as the result of the MBFSMTL method for heterogeneous clusters. The entire process is described in Fig. 5 using pseudo-code.

```
for (i = 0; i < nodes.count(); i++) {
  currentNumber = 0;
  currentSum = 0.0;
  setStateOfAllNodes(nodes, WHITE);
  setSubNetworkNumberOfAlNodes(nodes, 0);
  subNetworkWeight = subNetworkWeights[0];
  nodesToExplore = new Queue();
  currentNode = nodes[i];
  currentNode.state = GRAY;
  nodesToExplore.push(currentNode);
  while (!nodesToExplore.empty()) {
    currentNode = nodesToExplore.pop();
    neighbors = currentNode.neighbors();
    for (j = 0; j < neighbors.count(); j++) {
      neighbor = neighbors[j];
      if (neighbor.node.state == WHITE) {
        neighbor.node.state = GRAY;
        nodesToExplore.push(neighbor.node);
        currentSum += neighbor.weight;
      }
    }
    currentNode.state = BLACK;
    currentNode.subNetworkNumber = currentNumber;
    if (currentSum > subNetworkWeight) {
      currentNumber++;
      currentSum = 0;
      subNetworkWeight = subNetworkWeights[currentNumber];
    }
  }
  divisions[i] = storeCurrentDivision(nodes);
}
bestDivision = findBestDivision(divisions);
```

Fig. 5. The algorithm of the MBFSMTL method for heterogeneous clusters

5.5. GAMTL Method for Heterogeneous Clusters

As it was said in Section 4.3, the GAMTL method utilizes a genetic algorithm. Similar to the MBFSMTL method, the GAMTL method utilizes the road traffic network with traffic lanes assigned with weights and the portions of the load of the particular nodes as its inputs.

Similar to the GAMTL method for homogenous clusters, each individual is represented by a vector of integer values with length corresponding to the total number of crossroads *K*. Each individual corresponds to a single assignment of the crossroads to the particular sub-networks (see example for two sub-networks depicted in Fig. 6). The initial population of 90 individual is randomly generated. This means that, in each individual, the crossroads are randomly assigned to the particular sub-networks.

Fig. 6. An individual with corresponding assignment of the crossroads

Once the initial population is generated, the fitness function is calculated for each individual. Similar to the homogenous GAMLT method, the fitness functions consists of two parts – the *compactness* and the *equability*.

The compactness is used for minimization of the number of divided lanes. It can be calculated as:

$$C = \frac{L - L_D}{L}, \tag{5}$$

where C is the compactness, L_D is total number of divided lanes, and L is the total number of lanes. The compactness is calculated in the same way for both heterogeneous and homogenous clusters.

The only part of the GAMTL method, which is different for the heterogeneous clusters, is the calculation of the equability, which is used for correct distribution of the load among the sub-networks. For the heterogeneous clusters, the load of the road traffic network cannot be divided uniformly, but rather using the portions of the load of the particular nodes. Hence, the equability can be calculated as:

$$E = 1 - \frac{\sum_{i=1}^{M} \frac{\left| r_i - \frac{W_{Si}}{W_T} \right|}{\max(r_i, \frac{W_{Si}}{W_T})}}{M}, \tag{6}$$

where E is the equability, W_{Si} is the total weight of the ith sub-network using the assignment of the crossroads corresponding to the individual, for which the equability is calculated, W_T is the total weight of the road traffic network (see Equation (3)), M is the number of sub-networks, and r_i is the portion of the load of the node, on which the simulation of the ith sub-network will be performed.

With both compactness and equability calculated, it is possible to calculate the entire fitness function as:

$$F = r_E \cdot E + (1 - r_E) \cdot C, \tag{7}$$

where F is the fitness function, C is the compactness, E is the equability and r_E is the ratio of the equability, which can be set in range (0, 1>.

The r_E determines the preference of the compactness (optimization of the inter-process communication) or the equability (load-balancing of the sub-networks). For r_E equal to 0, the GAMTL method is focused solely on the compactness. So, the division with minimal number of divided lanes is found. However, the minimal possible number of divided lanes is zero, which happens in case that there is only one resulting sub-network (i.e. the traffic network is not divided at all). Since such a division is useless for our purposes, the r_E should be greater than 0. For r_E equal to 1, the GAMTL method is focused solely on the load-balancing and the number of divided lanes is not taken into account. This setting is theoretically usable, because the required number of sub-networks is created. Nevertheless, a very high number of divided lanes is generated, because there is no component of the fitness function, which would prevent it. Hence, an intense inter-process communication and consequently a non-negligible reduction of the speed of the distributed simulation must be expected. Based on preliminary experiments with the settings of the r_E, the default value is set to 0.25. With this setting, the number of divided traffic lanes is sufficiently low for many instances and the resulting sub-networks are still well load-balanced.

Once the fitness function is calculated for all 90 individuals, ten individuals with highest fitness function are selected for crossover using pairs of selected individuals. Each combination of two parent individuals produces two offspring (see Fig. 7). The first offspring receives integer values of even indices from the first parent and integer values of odd indices from the second parent. The second offspring receives all remaining values from both parents.

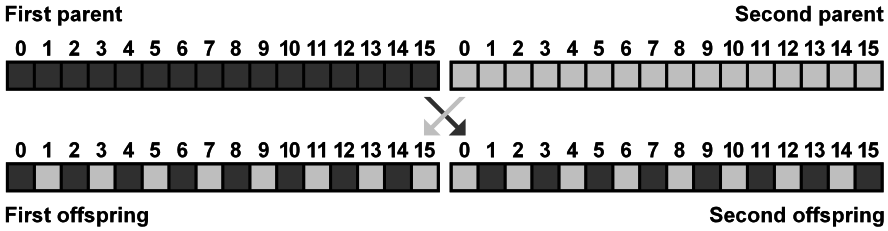


Fig. 7. The crossover of two parent individuals producing two offspring

The combination of all selected individuals produces a new generation of 90 individual. Each individual can be mutated. The mutation is limited to the maximum of five mutations per individual, which means that up to five values of each individual of the new generation can be randomly changed. Then, the fitness value is calculated for all individuals and the process repeats for 10 000 generations. All parameters of the GAMTL method for heterogeneous clusters (number of generations, number of individual, r_E, etc.) were adopted from the GAMTL method for homogenous clusters. There, the parameters were set based on preliminary testing of the method.

Once the genetic algorithm is completed, the traffic lanes connecting crossroads assigned to different sub-networks are marked as divided, similar to the MBFSMTL method.

6. Tests and Results

Both described methods for heterogeneous clusters have been thoroughly tested. There were two sets of tests. The first set of tests (see Section 6.2) was the benchmark of the heterogeneous cluster, on which the distributed road traffic simulation was tested (see Section 5.3). This benchmark was performed prior the division of the road traffic networks, since its result (portions of the load) is a necessary input for both division methods for heterogeneous clusters. In the second set of tests (see Section 6.3), both division methods for heterogeneous clusters were tested and compared to their counterparts for homogenous clusters.

6.1. Heterogeneous Cluster Used for Testing

Both sets of tests were performed on a heterogeneous cluster consisting of nine nodes. Eight nodes were used as working nodes performing simulation of particular road traffic sub-networks. The remaining (control) node served as a centralized barrier for the synchronization of the working processes (see Section 2.4). The parameters are summarized in Table 1.

Table 1. Parameters of the nodes of the distributed cluster for testing

No.	Type	CPU	RAM [MB]	OS
1	Working	Intel Celeron 1.70 GHz	504	WinXP SP3
2	Working	Intel Pentium D 3.6 GHz	512	Debian 5.0.1
3	Working	Intel Xeon 3.2 GHz	2048	Debian 5.0.1
4	Working	Intel Core 2 DUO 2.66 GHz	4096	WinXP SP3
5	Working	Intel Core i5-2400S 2.5 GHz	8192	Win7 64 bit
6	Working	Intel Xeon 3.2 GHz	2048	Debian 5.0.1
7	Working	Intel Xeon 3.2 GHz	2048	Debian 5.0.1
8	Working	Intel Pentium D 3.6 GHz	512	Debian 5.0.1
9	Control	Intel Xeon 3.2 GHz	2048	Debian 5.0.1

6.2. Benchmark of the Working Nodes of the Heterogeneous Cluster

The benchmark of the particular working nodes of the heterogeneous cluster was performed following the instructions described in Section 5.3. So, the results of the benchmark are the portions of the load of the particular nodes. In order to test the road traffic network division for various numbers of sub-

networks, various numbers (two, four, and eight) of working nodes were used (see Section 6.3). Hence, the portions of the load were calculated for two working nodes (working nodes 1-2, control node 9), four working nodes (working nodes 1-4, control node 9), and eight working nodes (working nodes 1-8, control node 9).

The benchmark was performed as follows. Sequential simulations of three road traffic networks (see Fig. 3) were performed ten times on each working node. Each simulation run was 900 time steps (corresponding to 15 minutes of the real time) long. Two values were observed in every simulation run – the mean number of vehicles moving within the road traffic network and the computation time of the simulation run. The computation times and the mean numbers of vehicles collected during ten simulation runs were then averaged and are summarized in Table 2 and Table 3, respectively.

As can be seen in Table 2, there are significant differences of the computation times of the same road traffic network simulated on different nodes. It can be observed that the computation time is not linearly dependent on the CPU frequency. Nevertheless, the results are consistent with the anticipated computational power of the particular nodes.

Table 2. The mean computation times of the simulation runs

	Number of crossroads of traffic network		
	16	64	256
Node No.	Computation time [ms]		
1	6 952	25 449	98 386
2	3 082	12 808	48 145
3	4 294	16 726	62 620
4	1 658	7 466	27 492
5	1 095	4 664	18 017
6	4 274	16 728	62 563
7	4 286	16 746	62 618
8	3 106	12 806	48 153

Lastly, it should be noted that the values for nodes 2 and 8 are very similar and the values for nodes 3, 6, and 7 are very similar as well. This is not a coincidence. The nodes 2 and 8 and the nodes 3, 6, and 7 are computers with identical parameters. So, it could be assumed that the results of the benchmark will be the same for all identical nodes. Nevertheless, in order to verify this assumption, the benchmark was performed on all nodes. This way, it was shown that the results for nodes 2 and 8 are nearly identical and the results for nodes 3, 6, and 7 are nearly identical as well (see Table 2). The maximal difference is less than 1 %.

As can be seen in Table 3, the mean number of vehicles moving within the road traffic network is not influenced by the node, on which the road traffic network is simulated. This is an expected and required behavior, since the road traffic simulation should give the same results regardless the computer, on which it is performed. The negligible differences of the particular values in

one column are caused by the stochastic nature of the simulation. Due to this nature, even two simulation runs of the same road traffic network performed on the same node are slightly different.

Table 3. The mean number of vehicles of the simulation runs

	Number of crossroads of traffic network		
	16	64	256
Node No.	Mean number of vehicles during simulation run		
1	724	1 660	3 514
2	721	1 636	3 585
3	726	1 647	3 585
4	720	1 684	3 514
5	718	1 666	3 535
6	727	1 649	3 580
7	723	1 652	3 593
8	717	1 641	3 581

Based on the obtained values, the computational power coefficients and the portions of the load of each node for two, four, and eight utilized working nodes can be calculated using the Equation (1) and Equation (2), respectively. The results are summarized in Table 4. The portions of the load of the particular nodes were used as the input for both division methods for heterogeneous clusters.

Table 4. Computational power coefficients and portions of the load

Nodes count	Node No.	Computational power coefficient	Portion of the load
2	1	0.0684	0.3199
	2	0.1454	0.6801
4	1	0.0684	0.1170
	2	0.1454	0.2487
	3	0.1083	0.1853
	4	0.2625	0.4491
8	1	0.0684	0.0507
	2	0.1454	0.1078
	3	0.1083	0.0803
	4	0.2625	0.1946
	5	0.4032	0.2989
	6	0.1086	0.0805
	7	0.1083	0.0802
	8	0.1444	0.1071

6.3. Performance of the Methods for Heterogeneous Clusters

The performances of the MBFSMTL and GAMTL methods for heterogeneous clusters were tested and compared to the performances of the MBFSMTL and GAMTL methods for homogenous clusters. Using this approach, it is shown that the consideration of the computational power of particular nodes of the heterogeneous cluster during the road traffic network division influences the resulting speed of the distributed road traffic simulation.

For the testing, four road traffic networks were used. Network 1 was an irregular road traffic network of 55 crossroads inspired by the Bory district of the Pilsen city, Czech Republic. Networks 2, 3, and 4 were regular square grids of 64, 256, and 1 024 crossroads, respectively. The networks were divided into two, four, and eight sub-networks using both methods for heterogeneous clusters and both methods for homogenous clusters. Then, the distributed microscopic road traffic simulation using the DUTS system was performed on two, four, and eight working nodes of the heterogeneous cluster (see Section 6.1), respectively.

For each combination of the divided road traffic network, the number of sub-networks, and the division method, ten simulation runs were performed and the computation time and the numbers of vehicles moving in particular sub-networks were observed. Similar to the benchmark, each simulation run was 900 time steps long, which corresponds to 15 minutes of the real time. This length of the simulation runs was selected, because the real data of the road traffic intensities in Pilsen city are collected in 15-minutes-long intervals. This enables an easier comparison of the results from the simulation with the real measured data in the future [27].

The averaged numbers of vehicles moving in particular sub-networks were used for investigation of the distribution of the total load among the particular sub-networks in the distributed road traffic simulation. The portions of the load of the particular sub-networks are compared to the intended values obtained from the benchmark (see Table 5 for the MBFSMTL and Table 6 for the GAMTL method).

As can be seen in Table 5 and Table 6, the real portions of the load of the particular sub-networks are similar (but not identical) to the intended values regardless the utilized division method. The mean differences between the intended and real portions are low for both methods – 8.5 % for the MBFSMTL method and 9.3 % for the GAMTL method. Hence, from this point of view, both methods give similar results. The reason for the observed differences between intended and real portions of the load is that the methods are not focused solely on load-balancing, but on the number of divided traffic lanes as well. Hence, both methods can produce a division, in which the load-balancing is not perfect, but the number of divided traffic lanes is low.

The averaged computation times of the distributed road traffic simulation are summarized in Table 7. As can be seen, both methods for heterogeneous clusters give better results (i.e. lower computation time of distributed simulation run) than both methods for homogenous clusters. The savings

vary from 3 % to 50 %. Such a high variance can be observed, because the total computation time of the distributed simulation is not influenced solely by the load-balancing, but by the inter-process communication as well.

Table 5. Portions of load of particular sub-networks created by MBFSMTL method

MBFSMTL method			Number of crossroads			
Nodes count	Node No.	Intended portion	55	64	256	1 024
			Portions of the load			
2	1	0.3199	0.2786	0.2521	0.2655	0.2513
	2	0.6801	0.7134	0.7679	0.7345	0.7487
4	1	0.1170	0.1376	0.0989	0.0991	0.1084
	2	0.2487	0.2218	0.2547	0.2669	0.2391
	3	0.1853	0.1950	0.2001	0.1741	0.1719
	4	0.4491	0.4456	0.4463	0.4599	0.4806
8	1	0.0507	0.0411	0.0551	0.0599	0.0479
	2	0.1078	0.1112	0.1189	0.0996	0.0981
	3	0.0803	0.0901	0.0761	0.0813	0.0816
	4	0.1946	0.2139	0.2144	0.2034	0.2017
	5	0.2989	0.2782	0.3007	0.3001	0.2916
	6	0.0805	0.0751	0.0697	0.0911	0.0722
	7	0.0802	0.0791	0.0718	0.0765	0.0709
	8	0.1071	0.1113	0.0933	0.0881	0.1360

Table 6. Portions of load of particular sub-networks created by GAMTL method

GAMTL method			Number of crossroads			
Nodes count	Node No.	Intended portion	55	64	256	1 024
			Portions of the load			
2	1	0.3199	0.3491	0.2992	0.2799	0.2751
	2	0.6801	0.6509	0.7008	0.7201	0.7249
4	1	0.1170	0.1118	0.1203	0.1021	0.0977
	2	0.2487	0.2193	0.2409	0.2764	0.2189
	3	0.1853	0.1604	0.2130	0.1919	0.2044
	4	0.4491	0.5085	0.4258	0.4296	0.4790
8	1	0.0507	0.0391	0.0611	0.0626	0.0433
	2	0.1078	0.0961	0.1109	0.1089	0.1189
	3	0.0803	0.0902	0.0727	0.0894	0.0788
	4	0.1946	0.2005	0.2011	0.2107	0.2119
	5	0.2989	0.2501	0.3107	0.2856	0.2999
	6	0.0805	0.1055	0.0788	0.0765	0.0911
	7	0.0802	0.0917	0.0809	0.0719	0.0688
	8	0.1071	0.1268	0.0838	0.0944	0.0873

The inter-process communication highly depends on the number of divided traffic lanes and this number varies significantly depending on the divided

road traffic network and the utilized division method. The numbers of divided traffic lanes are summarized in Table 8.

Table 7. Mean computation time of the simulation run

		Computation time [s]			
Nodes count	Crossroads count	Homogenous		Heterogeneous	
		MBFSMTL	GAMTL	MBFSMTL	GAMTL
2	55	29.7	25.5	22.9	21.1
	64	32.9	24.8	23.0	21.8
	256	66.3	66.5	58.3	54.2
	1 024	207.0	216.8	193.2	197.9
4	55	15.7	12.4	9.7	8.9
	64	19.0	12.1	11.5	9.5
	256	25.1	27.9	21.2	19.7
	1 024	76.9	96.6	72.6	74.0
8	55	8.9	8.1	6.3	6.0
	64	10.1	8.4	6.4	5.2
	256	16.2	16.9	13.2	12.5
	1 024	49.5	56.9	42.1	47.7

Table 8. Numbers of divided traffic lanes

		Number of divided traffic lanes			
Nodes count	Crossroads count	Homogenous		Heterogeneous	
		MBFSMTL	GAMTL	MBFSMTL	GAMTL
2	55	27	21	14	16
	64	44	16	16	18
	256	44	80	28	54
	1 024	108	196	82	144
4	55	70	30	63	27
	64	100	32	81	32
	256	170	142	134	108
	1 024	280	794	210	706
8	55	94	67	79	65
	64	144	116	98	80
	256	322	576	360	498
	1 024	592	1 112	588	996

From Table 8, it is clear that the GAMLT method for both homogenous and heterogeneous clusters generate a high number of divided traffic lanes for larger traffic networks (network 3 – 256 crossroads and network 4 – 1 024 crossroads). The reason for this is that, for a high number of crossroads, the genetic algorithm of the GAMTL method is unable to reach optimal division within the preset number of generations (i.e. 10 000 generations). So, the crossroads assigned to the particular sub-networks are not sufficiently clustered together. Consequently, many crossroads has crossroads from

different sub-networks as their neighbors. This greatly increases the number of divided traffic lanes and affects the resulting computation time of the distributed simulation.

In order to investigate this behavior further, another set of tests was performed. Two largest traffic networks (network 3 – 256 crossroads and network 4 – 1 024 crossroads) were divided into four sub-networks using the heterogeneous GAMTL method with 10 000, 100 000, and 1 000 000 generations, respectively. The best achieved compactness, equability, and fitness value, the computation time necessary for the division, and the numbers of divided traffic lanes were observed during the division. The division was performed on the node 4 of the heterogeneous cluster (see Table 1 for parameters). Then, the distributed road traffic simulation of the four resulting sub-networks was performed ten times for each combination of the network and the number of generations on working nodes 1 to 4 and control node 9 (see Table 1 for parameters) and the mean computation time was determined. The results are summarized in Table 9.

As can be seen in Table 9, the best achieved compactness, equability, and fitness value increase with increasing number of generations. Consequently, the quality of division is improving with increasing number of generations. This means that the number of divided traffic lanes is lower (corresponds to compactness) and the resulting sub-networks are better load-balanced (corresponds to equability). Nevertheless, this improvement is minimal (see last two rows of Table 9) and for the price of significant increase of computation time necessary for road traffic network division (see sixth row of Table 9).

Table 9. Dependency of the heterogeneous GAMTL method on the generations count

Crossroads	256			1 024		
Generations	10^4	10^5	10^6	10^4	10^5	10^6
Compactness	0.9007	0.9136	0.9265	0.8329	0.8381	0.8542
Equability	0.5777	0.7289	0.7445	0.7575	0.7615	0.7973
Fitness	0.8200	0.8674	0.8810	0.8150	0.8179	0.8310
Division time [s]	46.2	422.1	4 159.1	180.8	1 699.0	15 795.0
Divided lanes	108	94	80	706	684	616
Simulation time [s]	19.7	19.6	19.3	74.0	73.8	73.4

This slow increase of the quality of division corresponds to the result of the testing of the GAMTL method for homogenous clusters (see [29]). However, a better result could be expected from both homogenous and heterogeneous versions of the GAMTL method. The reason for the slow increase of the quality of division is probably suboptimal settings of the parameters of the utilized genetic algorithm. The most obvious parameters are the number of individuals in a generation, number of selected individuals from each generation, number of mutations, and the value of the r_E. These parameters

were set using preliminary testing of the homogenous GAMTL method and were adopted by the heterogeneous version. The preliminary testing suggested that the setting was convenient. Nevertheless, for optimal settings, a more thorough testing of all combinations of the particular parameters would be required. Moreover, there are other features of the genetic algorithm, which can be also very important (e.g. type of crossover, type of selection, etc.). These features have not been investigated yet.

The optimal settings of all mentioned parameters and features of the genetic algorithm of the GAMTL method is outside the scope of this paper. Nevertheless, it is one of the main aims of our future work. More information can be found in [30].

7. Conclusions

In this paper, we have described two methods for division of road traffic networks for heterogeneous clusters – the MBFSMTL and the GAMTL. Both these methods are based on their counterparts originally designed for homogenous clusters. The methods for heterogeneous clusters divide the load among the particular road traffic sub-networks using a ratio based on the computational powers of the particular nodes of the heterogeneous cluster. In order to calculate the ratio, all nodes of the target heterogeneous cluster are investigated for their computational powers using a benchmark test.

The performances of both methods for heterogeneous clusters were thoroughly tested and compared mutually and with their counterparts for homogenous clusters. Both methods for heterogeneous clusters showed better results then both methods for homogenous clusters for all tested instances. The savings of the computational time of the distributed simulation reached up to 50 %. Hence, it is clear that the adaptation of the load of particular sub-networks for the computational power of the nodes has a significant effect on the resulting computation time of the distributed simulation.

When both methods for heterogeneous clusters are compared together, the GAMTL method gives better results for smaller road traffic networks than the MBFSMTL methods. On the contrary, the MBFSMTL method gives better results for larger traffic networks, and shows more stable results due to the lower number of divided traffic lanes. At first glance, this makes the MBFSMTL method more utilizable than the GAMTL method. However, the worse results of the GAMTL method could be caused by suboptimal settings of the parameters of the utilized genetic algorithm. So, it is possible that, after an optimization, the GAMTL method will give better results than the MBFSMTL method.

In our future work, we will first focus on the optimization of the GAMTL method for both homogenous and heterogeneous clusters in order to achieve lower number of divided traffic lanes even for large traffic networks.

Another step in our research is the combination of the described methods for the load-balanced division of road traffic networks with efficient communication protocols. For the testing of the division methods described in this paper, a basic (i.e. not optimized) communication protocol was used. However, we have developed several advanced communication protocols, which can significantly reduce the amount of inter-process communication [3]. Since the inter-process communication is relatively slow, the utilization of an advanced communication protocol can significantly improve the overall performance of the distributed road traffic simulation. Moreover, some of the advanced protocols could mitigate the higher number of divided traffic lanes produced by the division methods in some instances [3].

Another promising direction of our research is the parallel/distributed road traffic simulation, which can better exploit the computational powers of the nodes with multiple or multi-core processors than a pure distributed simulation due to the utilization of shared memory instead of message passing were possible [12].

References

1. Fujimoto, R. M.: Parallel and Distributed Simulation Systems. John Wiley & Sons, New York, USA. (2000)
2. Lighill, M. H., Whitman, G. B.: On kinematic waves II: A theory of traffic flow on long crowed roads. In Proceedings of the Royal Society of London, s. A, 229, London, United Kingdom. (1955)
3. Potuzak, T.: Methods for Reduction of Interprocess Communication in Distributed Simulation of Road Traffic. Doctoral thesis, University of West Bohemia, Pilsen, Czech Republic. (2009)
4. Nagel, K., Schreckenberg, M.: A Cellular Automaton Model for Freeway Traffic. In Journal de Physique I, 2, 2221–2229. (1992)
5. Gipps, P. G.: A behavioural car following model for computer simulation. In Transp. Res. Board, 15-B(2), 403–414. (1981)
6. Nagel, K., Rickert, M.: Parallel Implementation of the TRANSIMS Micro-Simulation. In Parallel Computing, Vol. 27, No. 12, 1611–1639. (2001)
7. Gonnet, P. G.: A Queue-Based Distributed Traffic Micro-simulation. Technical report. (2001)
8. Potuzak, T.: Methods for Division of Road Traffic Networks Focused on Load-Balancing. In Advances in Computing, Vol. 2, No. 4, 42–53. (2012)
9. Nizzard, L.: Combining Microscopic and Mesoscopic Traffic Simulators. Raport de stage d'option scientifique, Ecole Polytechnique, Paris, France. (2002)
10. Nagatani, T.: Gas Kinetic Approach to Two-Dimensional Traffic Flow. In J. Phys Soc Jap, Vol. 65, No. 10, 3150–3152. (1996)
11. Burghout, W.: Hybrid microscopic-mesoscopic traffic simulation. Doctoral thesis, Royal Institute of Technology, Stockholm, Sweden. (2004)
12. Potuzak, T.: Distributed-Parallel Road Traffic Simulator for Clusters of Multi-core Computers. In 2012 IEEE/ACM 16th International Symposium on Distributed Simulation and Real Time Applications – DS-RT 2012, Dublin, Ireland, 195–201. (2012)

13. Klein, U., Schulze, T., Strassburger, S., Menzler, H.: Distributed Traffic Simulation Based on the High Level Architecture. In Proceedings of Simulation Interoperability Workshop, Orlando, USA. (1998)
14. Kiesling, T., Lüthi, J.: Towards Time-Parallel Road Traffic Simulation. In Proceedings of the Workshop on Principles of Advanced and Distributed Simulation. (2005)
15. Potuzak, T.: Division of Traffic Network for Distributed Microscopic Traffic Simulation Based on Macroscopic Simulation. In Proceedings of the 7th EUROSIM Congress on Modelling and Simulation, Vol. 2, Prague, Czech Republic. (2010)
16. Barcelo, J., Ferrer, J. F., Garci, D., Florian, M., Le Saux, E.: The Parallelization of AIMSUN2 Microscopic Simulator for ITS Applications. In Proceedings of 3rd World Congress on Intelligent Transportation Systems, Orlando, Florida. (1996)
17. Hartman, D.: Leading Head Algorithm for Urban Traffic Model. In Proceedings of the 16th International European Simulation Symposium ESS, Budapest, Hungary, 297–302. (2004)
18. Cetin, N., Burri, A., Nagel, K.: A Large-Scale Agent-Based Traffic Microsimulation Based on Queue Model. In Proceedings of 3rd Swiss Transport Research Conference, Monte Veritas. (2003)
19. Klefstad, K., Zhang, Y., Lai, M., Jayakrishnan, R., Lavanya, R.: A Scalable, Synchronized, and Distributed Framework for Large-Scale Microscopic Traffic Simulation. In The 8th International IEEE Conference on Intelligent Transportation Systems. (2005)
20. Wei, D., Chen, W., Sun, X.: An Improved Road Network Partition Algorithm for Parallel Microscopic Traffic Simulation. In 2010 International Conference on Mechanic Automation and Control Engineering, 2777–2782, Wuhan, China. (2010)
21. Potuzak, T.: Usability of Macroscopic and Mesoscopic Road Traffic Simulations in Division of Traffic Network for Distributed Micro-scopic Simulation. In CSSim 2011 – Conference on Computer Modelling and Simulation, Brno, Czech Republic, 94–101. (2011)
22. Potuzak, T.: Comparison of Road Traffic Network Division Based on Microscopic and Macroscopic Simulation. In UKSim 2011 – UKSim 13th International conference on Computer Modelling and Simulation, Cambridge, United Kingdom, 409–414. (2011)
23. Knuth, D. E.: The Art of Computer Programming Vol. 1. 3rd edition, Addison-Wesley. (1997)
24. Farshbaf, M., Feizi-Darakhshi, M.: Multi-objective Optimization of Graph Partitioning using Genetic Algorithms. In 2009 Third International Conference on Advanced Engineering Computing and Applications in Sciences, Sliema. (2009)
25. Menouar, B.: Genetic Algorithm Encoding Representations for Graph Partitioning Problems. In 2010 International Conference on Machine and Web Intelligence (ICMWI), Algiers, 288–291. (2010)
26. Potuzak, T.: Utilization of a Genetic Algorithm in Division of Road Traffic Network for Distributed Simulation. In ECBS-EERC 2011 – 2011 Second Eastern European Regional Conference on the Engineering of Computer Based Systems, Bratislava, Slovakia, 151–152. (2011)
27. Potuzak, T.: Division of Road Traffic Network for Distributed Simulation Performed on Heterogeneous Clusters. In ECBS 2012 – 2012 IEEE 19th International Conference and Workshops on Engineering of Computer-Based Systems, Novi Sad, Serbia, 117–125. (2012)

28. Bohna, C. A., Lamontb, G. B.: Load Balancing for Heterogeneous Clusters of PCs. In Future Generation Computer Systems, Vol. 18, No. 3, 389–400. (2002)
29. Potuzak, T.: Suitability of a Genetic Algorithm for Road Traffic Network Division. In KDIR 2011 – Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Paris, France, 448–451. (2011)
30. Potuzak, T.: Issues of Optimization of a Genetic Algorithm for Traffic Network Division using a Genetic Algorithm. In KDIR 2012 – Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Barcelona, Spain, 340–343. (2012)

Tomas Potuzak was born in 1983 in Sušice, Czech Republic. He went to University of West Bohemia (UWB) where he studied software engineering and obtained his degree in 2006. Then, he entered Ph.D. studies at the Department of Computer Science and Engineering (DCSE) at the same university and has worked on issues of distributed simulation of road traffic. He obtained his Ph.D. in 2009. He is now a senior lecturer at the DCSE UWB. His research is focused on the issues of distributed simulations and component-based simulations.

Received: June 01, 2012; Accepted: November 12, 2012.

DOI: 10.2298/CSIS120601007A

Modeling and Visualization of Classification-Based Control Schemes for Upper Limb Prostheses

Andreas Attenberger[1] and Klaus Buchenrieder[2]

[1] Institut für Technische Informatik
Universität der Bundeswehr München
Neubiberg, Germany
Andreas.Attenberger@unibw.de
[2] Institut für Technische Informatik
Universität der Bundeswehr München
Neubiberg, Germany
Klaus.Buchenrieder@unibw.de

Abstract. During the development of control schemes for upper-limb prostheses, the selection of a classification method is the decisive factor on predicting the correct hand movements. This contribution brings forward an approach to validate and visualize the output of a chosen classifier by simulative means. Using features extracted from a collection of recorded myoelectric signals (MES), a training set for different classes of hand movements is produced and validated with additional MES recordings. Using the output of the classifier, the behavior of an actual prosthesis is simulated by controlling the 3D model of a prosthetic hand. For systematic comparison of feature sets and classification methods, a toolbox for MATLAB™ has been developed. Our classification results show, that existing classification schemes based on EMG data can be improved significantly by adding NIR sensor data. Employing only two combined EMG-NIR sensors, five motion classes comprising full movements, including pronation and supination, can be distinguished with 100% accuracy.

Keywords: Classification Algorithms, Decision Trees, Electromyography, Modeling, Prosthetic Hand, Simulation, Support Vector Machines, Visualization.

1. Introduction

Research on the employment of myoeletric signals for prostheses control has been conducted since the 1940s [13]. Myoelectric signals (MES) can be measured by placing electrodes on the skin located over the observed muscles. When a muscle is activated through a neurological impulse, transmitted from the brain, small changes in electrical potential can be detected on the surface of the skin. In order to actuate a prosthesis, these signals are processed. In their work, Englehart, Hudgins, Parker and Stevenson provide a definition for the signal-classification problem and preposition a multi-stage process [4]. In this process, the complexity of recorded data is reduced by the introduction of

features like root-mean-square (RMS) values, which denote the average signal strength. After extraction, the selected features are fed into a classifier. Usually, a training-data set, with different classes for various hand movements or hand-positions, is created. Any new electromyographic (EMG) data can then be attributed to one of the given classes. Recently, research about a novel type of sensor using near-infrared (NIR) light, to detect muscle activity, has been disclosed [7] [6]. Near-infrared light is partially absorbed by the hemoglobin in the red blood cells. Due to this effect, different levels of absorption can be recorded using a NIR light source and a photodetector. As a result, the level of muscular activity in the area under the sensor can be observed and hand-positions as well as -movements detected.

In this contribution, we present a model of the classification process for upper-limb prostheses including subsequent simulation, validation and visualization. From the recorded sensor signals, RMS and zero crossing (ZC) features as well as a feature derived from the sensor's NIR component are extracted for five different hand movements. For training and classifier validation two different classification methods are demonstrated and compared. Both, an easy to implement decision tree algorithm as well as a more flexible multi-class support vector machine (SVM) are presented. For the simulation of this process, a 3D model of a hand prosthesis, as shown in Fig. 1, is employed for visualizing the classification results. This modeling and simulation solution is an example of the functionality offered by a custom-built MATLAB™ toolbox allowing the selection of features and the structured comparison of various classification methods for a faster evaluation of prosthesis control models. Furthermore, integrating additional information from NIR sensors leads to improved classification results. Over the years, an important factor in increasing classification accuracy for a higher number of hand movements has been the utilization of additional sensors [11]. However, achieving high accuracy for detecting four or more movement classes with only two sensors placed on the forearm remains challenging. Liu and Luo have built a classifier based on wavelet packet transformation and

Fig. 1. 3D Hand Prosthesis Model.

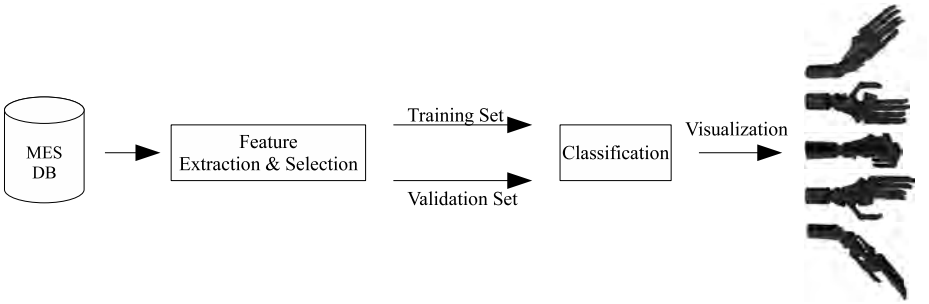


Fig. 2. Modeling, validating and visualizing the classification process.

a neural network (NN) that attains a detection rate of 98% for four hand movements [10]. Arvetti, Gini and Folgheraiter employ wavelet analysis and an NN to reach almost 97% accuracy for five different motion classes [1]. León, Leija and Muñoz identifiy seven different movements through a combination of discrete Fourier transformation and a NN with a success rate of up to 95% [9]. Note, that the last two methods only use either the first or the last part of the signal for identifying a class and not the full, transient movement. Additionally, only León, Leija and Muñoz include pronation and supination motion classes.

2. Method

This section describes our method of modeling, validation and visualization of the prosthesis control scheme. First of all, only employing EMG data, classification of five different hand movements is demonstrated for two different feature combinations. The features extracted from our database of hand movement recordings are used to train a SVM and a decision tree classifier, for which the results are subsequentially validated. Combining EMG and NIR sensor signals offers a significant improvement of the accuracy of a classifier. The final classification results are then used to control the visualization model of the prosthesis embedded in MATLAB™, as shown in Fig. 2.

2.1. Data Acquisition and Feature Extraction

For our investigation, we recorded 100 datasets for five different hand movements each comprising 20 data samples each. For every movement, the signals from two combined EMG and NIR sensors [7] were captured from the forearm of a proband. The sensors were placed over the extensor digitorum and the carpi radialis muscles. The data were recorded with a custom sensor system, integrating both, a single differential EMG sensor as well as a LED and a photo receiver for capturing the amount of near infrared light not absorbed in the underlying tissue.

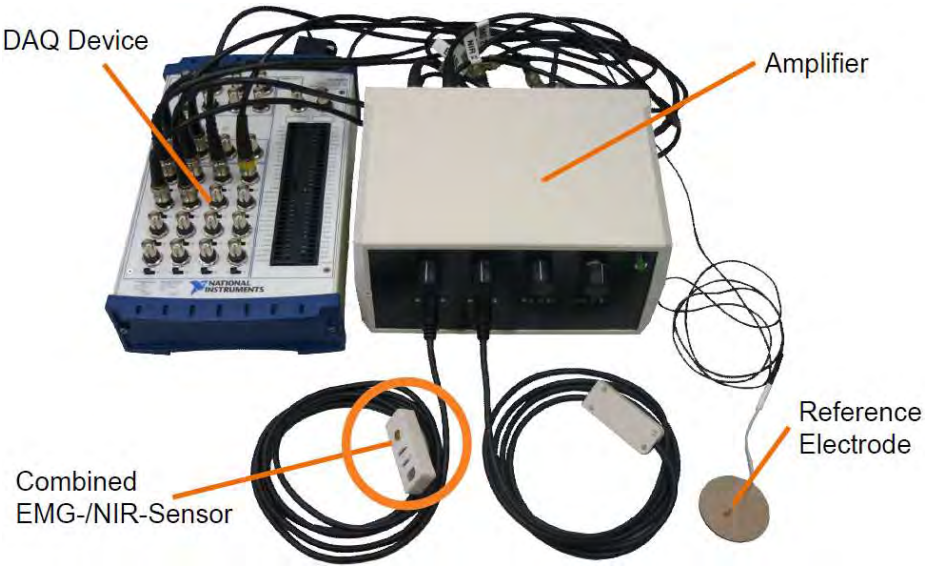


Fig. 3. The hardware used for acquiring EMG and NIR signals, including the DAQ and the sensor system.

The MES were amplified by a factor of about 10 dB. To prevent tissue damage from excessive heat, the NIR light emitted by the diode placed on the sensor is pulse modulated with a pulse rate of 16 Hz and a duty cycle of 2.5%. The rise and fall time of the pulses was 5%. For reducing interference between sensors, an offset of 15% was introduced. The enable signals for the pulses were generated by the MATLAB™ signal generator application displayed in Fig. 4 and output with a NI USB-6229 DAQ device from National Instruments. Fig. 3 shows the hardware setup necessary for acquiring the combined EMG and NIR signals. The EMG/NIR sensor consists of a single differential EMG electrode located between the NIR LED and the photo receiver. The sensors as well as a reference electrode connect to the main signal amplifier. The amplified analog signals are fed into the NI USB Device. Additionally, the enable signal output is also recorded with the DAQ device for further reference. The recordings were conducted with a frontend application created in MATLAB™ and Simulink™ using a sampling rate of 4096 Hz. Each five second data sample is enriched with a time-synchronous video recording of the proband's hand motion. The resulting data was saved in MATLAB™ binary files with the EMG and NIR recordings captured in arrays.

The first step towards creating a training set for the classifiers is the feature extraction from a myoelectric and a near-infrared signal. By this measure, the amount of input data and the complexity is reduced prior to the classification process [2]. Various features like the RMS, ZC or the waveform length can be extracted from EMG signals [12]. In order to gauge the strength of the MES for

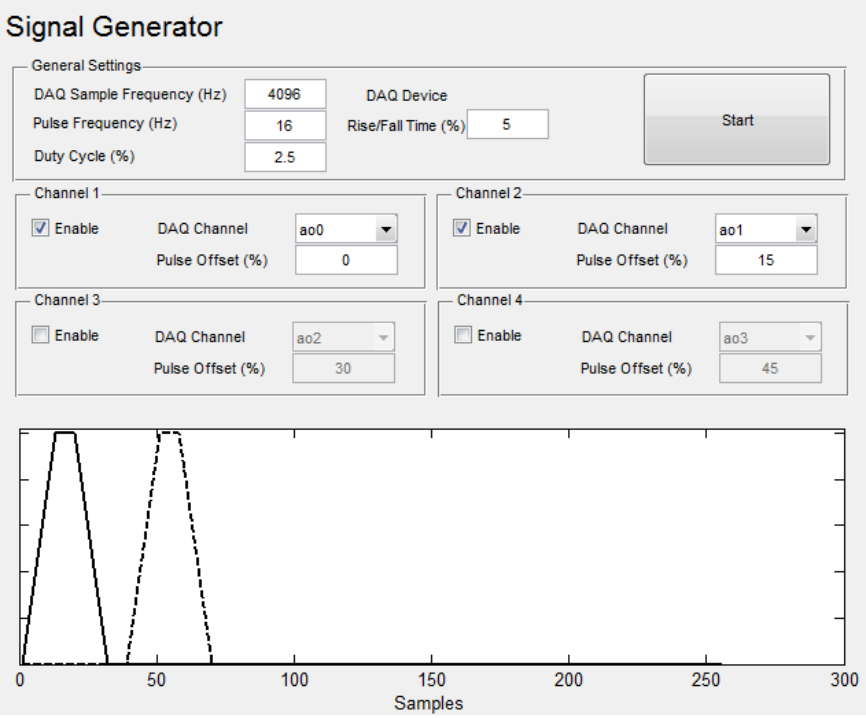


Fig. 4. Generator application for controlling the NIR sensor LEDs.

a set of N samples, the RMS results from:

$$x_{rms} = \sqrt{\frac{1}{N} \cdot \sum_{n=1}^{N} (x_n^2)} \,. \tag{1}$$

The window size was set to 256 samples with an increment of 256. The RMS feature was also used for preprocessing the MES recordings. Several seconds of noise recorded before and after the actual hand movement were removed by amplitude threshold provisory clipping. This was realized by measuring the RMS values of noise recorded with the EMG signal. All recordings contained at least one second of noise before the start of the movement. The first second of each recording was split into four windows of 256 samples and the RMS value of each window was calculated. The maximum out of these four results was then compared to a sliding window of 1024 samples throughout the remaining recording. A window with an RMS value higher than that of the maximum noise RMS sample window was considered to contain the start of the movement. Finally, a window with a resulting RMS equal to or lower than the RMS noise threshold was assumed to mark the end of the movement. The recordings from all sensors for a single recording were trimmed to preserve the integrity of the

EMG signals. Fig. 5 shows the first second of the EMG signal containing noise. Brackets shown beneath the signal denote the actual movement signal as well as the last 1024 sample window discarded due to its RMS value below the noise threshold. The signals from the EMG and NIR feature combination were treated accordingly with the NIRS feature used for determining the beginning and end of a movement as the NIR sensor signal yields a lower noise-to-signal-ratio. Window sizes were adjusted for four values per second, i.e., a 256 window size and increment for both the RMS and the NIRS.

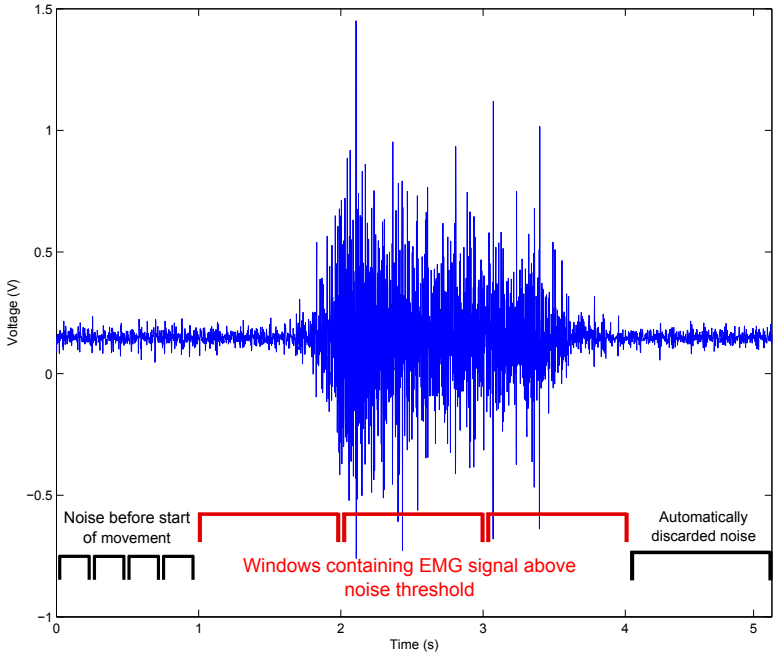


Fig. 5. A raw EMG signal recording with four 256-sample-windows used for determing the noise RMS and 1024-sample-windows for comparing the significant and ineffectual section.

In addition to the RMS, the ZC feature was also extracted from the EMG signal using:

$$x_{zc} = \sum_{n=0}^{N-1} I\{\mathrm{sgn}(n+1) \cdot \mathrm{sgn}(n) < 0\}\,. \tag{2}$$

As the LED is switched on periodically, a NIRS feature taking into account the pulse rate and duty cycle can be derived [6]. For the vectors $\overline{n} = \{n_1, ..., n_k\}$,

consisting of the measured NIR signal in the observed time frame, and $\overline{e} = \{e_1, ..., e_k\}$, with $Ena(e_i)$ denoting the state of the enable signal (either 0 or 1 depending on an upper and lower threshold) at each point of time in the signal window, the NIRS feature can be calculated as follows:

$$\text{NIRS} = \text{Signal}(\overline{n}, \overline{e}) - \text{Offset}(\overline{n}, \overline{e})\,. \tag{3}$$

with

$$\text{Signal}(\overline{n}, \overline{e}) = \frac{\sum_{i=1}^{k} n_i \cdot Ena(e_i)}{\sum_{i=1}^{k} Ena(e_i)}\,. \tag{4}$$

$$\text{Offset}(\overline{n}, \overline{e}) = \frac{\sum_{i=1}^{k} n_i \cdot (1 - Ena(e_i))}{\sum_{i=1}^{k} (1 - Ena(e_i))}\,. \tag{5}$$

To produce window sizes of equal length, for which the EMG and the NIRS features can be combined, the NIRS window size and its increment was set to 256 samples as well. For the combined features, the NIRS feature was chosen as the source for the amplitude threshold provisory clipping. This is advantageous because the NIRS feature clearly indicates the beginning of a muscle contraction, revealing the motion of a hand with more precision than RMS alone. Fig. 6 shows the DC corrected EMG signal and the derived RMS and ZC features as well as the NIRS feature from a sensor placed over the extensor digitorum during wrist extension. The RMS, ZC and NIRS features were calculated for a window size and increment of 256 samples.

Besides using a combination of individual features from the different signal types, the combined EMG and NIR sensor also offers the possibility of using a single feature integrating both the EMG as well as the NIR signal. One example is the NIRSRMS feature resulting from combining both the aforementioned NIRS as well as the RMS feature with the myoelectric signal $\overline{m} = \{m_1, ..., m_k\}$:

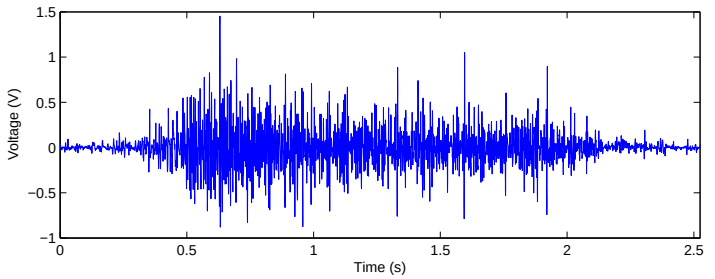
$$\text{NIRSRMS} = \text{RMS}(\overline{m}) \cdot \text{NIRS}(\overline{n}, \overline{e})\,. \tag{6}$$

Apart from the NIRSRMS combination, other features like the DC corrected NIRS signal – useful for realtime control – can be calculated from the NIR sensor data [6].

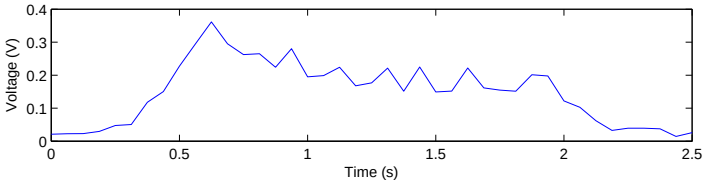
2.2. Training Set Creation

Out of the 20 data samples for each hand movement, 13 are drawn for training the classifier while the remaining 7 are deployed for the validation of the classification method. In the examplary classification process, we distinguish five different hand motions: fist, supination, pronation, wrist extension and wrist flexion.

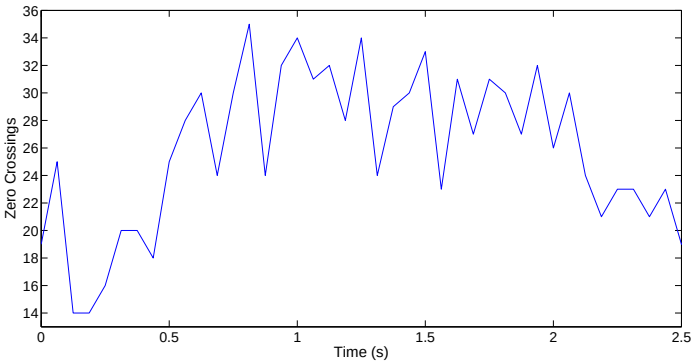
As a first classification method, we have chosen the decision tree algorithm from the MATLAB™ statistics toolbox. For each node t of the tree, a subset X_t



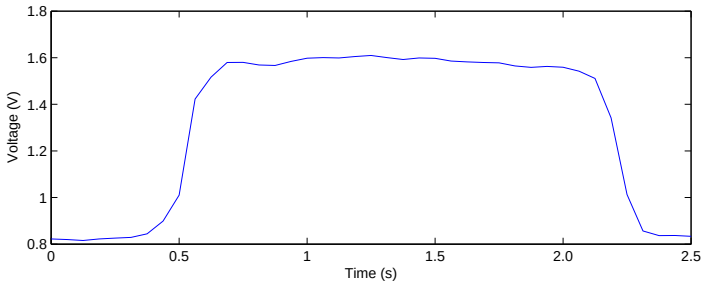
(a) EMG Signal



(b) RMS Feature derived from EMG



(c) ZC Feature derived from EMG



(d) NIRS Feature

Fig. 6. The EMG, EMG-calculated RMS, ZC and NIRS signal values for the extensor digitorum muscle during wrist extension.

is associated with it [14]. The subset is then split into two subsets for the descendant node, containing the 'Yes'-answers X_{tY} and the 'No'-answers X_{tN} for the question associated with the current node. The subsets satisfy:

$$X_{tY} \cap X_{tN} = \emptyset \,. \tag{7}$$

$$X_{tY} \cup X_{tN} = X_t \,. \tag{8}$$

The default splitting criterion used in MATLAB™ is the diversity index introduced by Gini for a node τ [8]:

$$i(\tau) = \sum_k p(k \mid \tau)^2 \,. \tag{9}$$

Altering the splitting criterion to other choices, offered by MATLAB™, did not yield a substantial increase in classification accuracy. Decision tree algorithms can quickly be implemented as the parameters are not critical. We have also investigated Support Vector Machines (SVM), which offer more flexibility. SVMs are linear classifiers which separate classes by means of hyperplanes. For a binary SVM, the hyperplane for a set of feature vectors x_i, with $i = 1, 2, ..., n$, which belong to the two classes ω_1 and ω_2, is denoted by [14]:

$$g(x) = \omega^T \cdot x + \omega_0 = 0 \,. \tag{10}$$

Multi-class SVMs can be constructed from binary SVMs by breaking up the original multi-class problem into several binary class problems [14]. The LIBSVM package employs the one-vs-one approach [3]. Depending on the type of training data, kernel choice and regularization constant can have an impact on the classification results of a SVM. Instead of a linear kernel, the authors of LIBSVM recommend the implemented RBF kernel, with:

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2} \,. \tag{11}$$

A five-fold cross-validation was used for finding a suitable value for the regularization and γ parameters. To demonstrate the effects of feature selection and the choice of the classifier, the previously described classification algorithms were applied to different feature sets. Starting with a combination of the RMS values, extracted from the MES recording of the extensor digitorum muscle and the ZC feature derived from the sensor placed over the flexor carpi radialis, the feature values were first subjected to DC correction and noise reduction. All data points below a threshold of 0.15 for the normalized RMS and 0.68 for the normalized ZC feature set were removed before creating the classifier training sets. With these sets, models were generated for the training phase of the SVM and the decision tree classifier. Fig. 8 and Fig. 7 depict the partitioning of the source data into five classes. However, using the RMS-ZC combination, the classifiers cannot unambiguously distinguish between the five movements. While this feature combination is sufficiently distant for most of the classes,

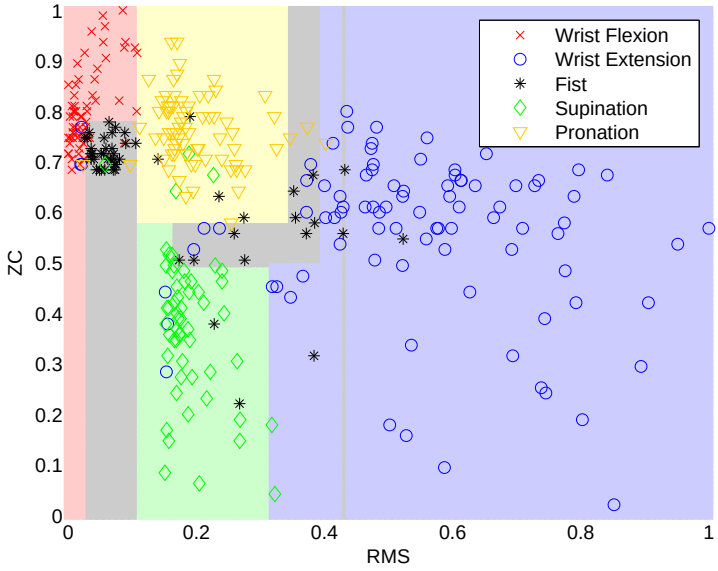


Fig. 7. Decision tree training set with 13 data samples for each class and combination of RMS and ZC features.

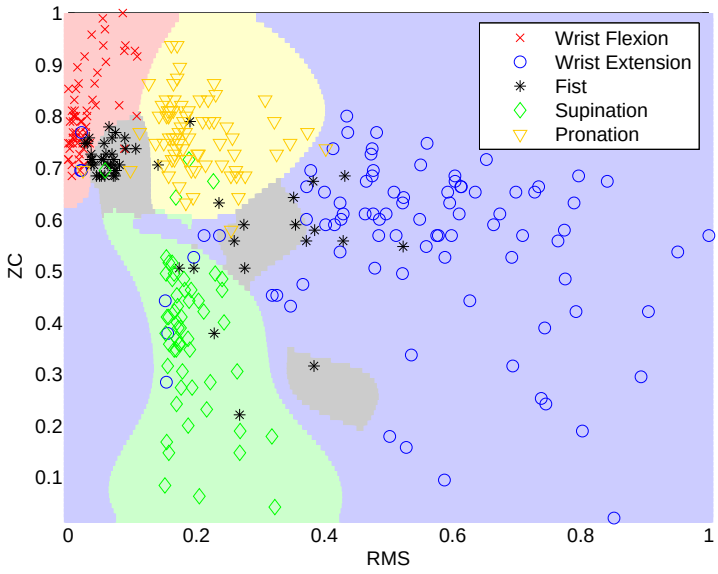


Fig. 8. SVM training set with 13 data samples for each class and a combination of RMS and ZC features.

closing a fist is not as clearly separated from neighboring classes such as wrist flexion or wrist extension.

In order to achieve better results, it is necessary to employ a different combination of features. Applying the same parameters for classifier training, Fig. 9 and Fig. 10 illustrate the results for the five classes with the RMS feature extracted from the data-sets for both the extensor digitorum and the flexor carpi radialis muscles. As evident from the location of the data points, this combination of features yields a clearer separation of the five hand movements. In this case - apart from the DC correction - additional noise reduction did not offer any further improvement of classification results.

Finally, for achieving even better distance between the data points of the motion classes, the RMS-RMS feature set was enriched with the NIRS feature data from both sensors, yielding a four dimensional feature space. Before training, both NIRS and RMS features were DC corrected. Then, the SVM and decision tree classifiers were trained again with this extended feature combination. The training models can now serve as reference for further classification. A validation as well as a visualization of the recording data is presented in the following section.

2.3. Validation and Visualization

In order to validate the classifier and its reference model, seven recordings of each hand movement were fed into the feature extraction process using the EMG and NIRS data. The derived features were then classified using the previously generated decision tree and the SVM models. Based on reference signals of known movements, classifier results were compared and validated. Table 1 contains the percentages of correctly identified hand movements for each class and the overall classification accuracy for the SVM and decision tree training models utilizing a RMS-ZC feature combination. Comparing the result of the RMS-ZC feature combination with the RMS-RMS combination in Table 2, the impact of feature selection prior to classifier training is confirmed. The validation results show an improvement between the RMS-ZC and RMS-RMS. Furthermore, the choice of the classification algorithm can have a substantial effect on accuracy as shown in Tables 1 and 2. Depending on the feature set, the simple decision tree algorithm may produce a variety of results, while the SVM classifier is more consistent. For this, parameters must be determined by cross validation in the training phase and initially set. Apart from classifier selection, our validation data demonstrates the value of the newly developed EMG-NIR sensor. In case of the selected five hand movement classes, 100% classification accuracy can be achieved by combining the recorded EMG and NIR data as presented in Table 3.

Finally, the simulation of a 3D-hand-model of a prosthesis was controlled with the classifier output. The visualization of a prosthesis is based on a model originally created in Autodesk™ 3ds Max [5]. The virtual hand, shown in Fig. 12, consists of components including shaft, wrist and joints for individual fingers as found in typical prostheses. For reduced complexity and better performance

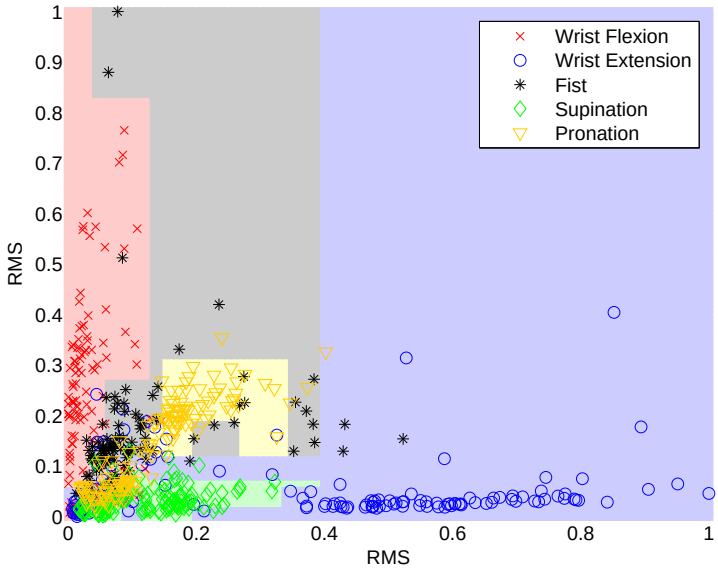


Fig. 9. Decision tree training set with 13 data samples for each class with two RMS features.

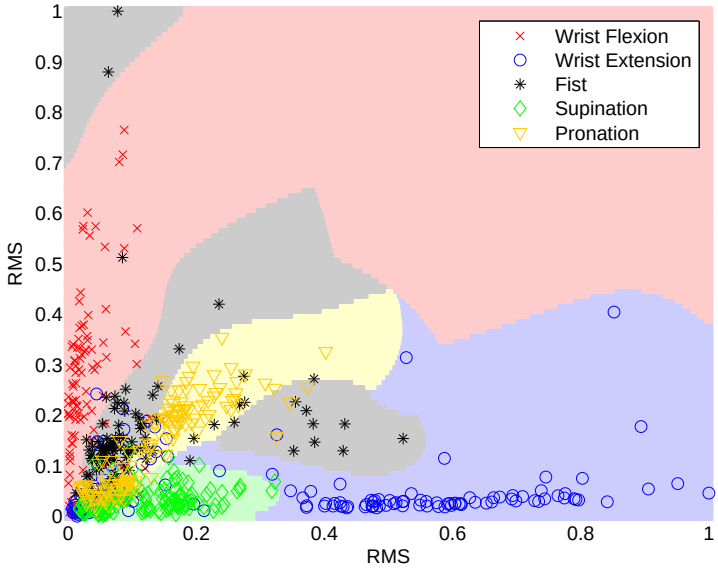


Fig. 10. SVM training set with 13 data samples for each class with two RMS features.

Table 1. Percentages of correct hand movements for the RMS-ZC feature set.

Hand Movement	SVM Model	Decision Tree Model
Wrist Flexion	85.7%	71.4%
Wrist Extension	100.0%	100.0%
Fist	57.1%	14.3%
Supination	100.0%	28.6%
Pronation	100.0%	100.0%
False Positives	2.9%	11.4%
False Negatives	8.6%	25.7%
Overall Accuracy	88.6%	62.9%

Table 2. Percentages of correct hand movements for the RMS-RMS feature set.

Hand Movement	SVM Model	Decision Tree Model
Wrist Flexion	100.0%	85.7%
Wrist Extension	57.1%	100.0%
Fist	100.0%	71.4%
Supination	100.0%	85.7%
Pronation	100.0%	100.0%
False Positives	8.6%	8.6%
False Negatives	0.0%	2.9%
Overall Accuracy	91.4%	88.6%

Table 3. Percentages of correct hand movements for the RMS-RMS-NIRS-NIRS feature set.

Hand movement	SVM Model	Decision Tree Model
Wrist Flexion	100.0%	0.0%
Wrist Extension	100.0%	100.0%
Fist	100.0%	100.0%
Supination	100.0%	100.0%
Pronation	100.0%	100.0%
False Positives	0.0%	20.0%
False Negatives	0.0%	0.0%
Overall Accuracy	100.0%	80.0%

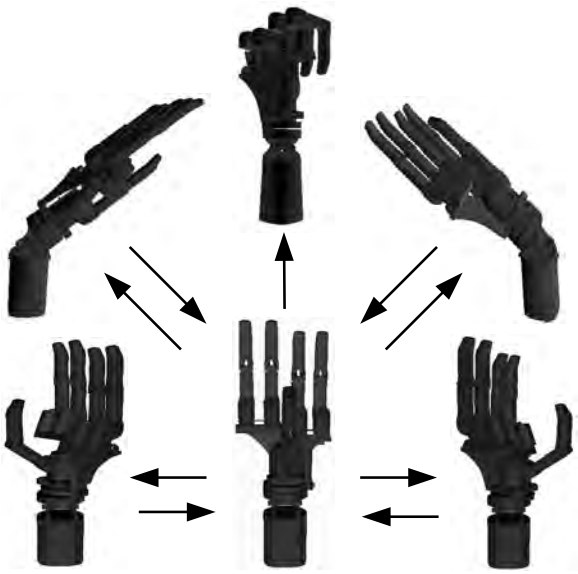


Fig. 11. 3D display of five hand movements (clockwise from bottom left to bottom right) starting and ending in the relaxed hand position (bottom center): pronation, wrist flexion, fist, wrist extension, supination.

during simulation, each finger consists of only two joints connected to a plate mounted on a rotary joint. Extending and flexing the 3D hand is realized with a pivoted joint at the base of the hand.

Fig. 12. The individual components of the 3D prosthesis hand model.

Listing 1.1. Setting hand positions for the 3D prosthesis and working with position files.

```
1  vp = VirtualProsthesis('prosthesis.WRL')
2
3  vp = vp.SetJointPosition('middle01', 90)
4  vp = vp.SetJointPosition('middle02', 90)
5  vp = vp.SetJointPosition('ring01', 90)
6  vp = vp.SetJointPosition('ring02', 90)
7  vp = vp.SetJointPosition('small01', 90)
8  vp = vp.SetJointPosition('small02', 90)
9  vp = vp.SetJointPosition('index01', 90)
10 vp = vp.SetJointPosition('index02', 90)
11 vp = vp.SetJointPosition('thumb01', 90)
12 vp = vp.SetJointPosition('thumb02', 90)
13
14 pose_fist = vp.GetHandPosition('fist')
15 vp = vp.SaveHandPosition(pose_fist)
16 vp = vp.WriteHandPositionsToFile('positions.mat')
17
18 vp = vp.ReadHandPositionsFromFile('positions.mat')
19 vp.GetSavedPositionNames()
20 vp = vp.LoadHandPosition('fist')
```

After conversion to the Virtual Reality Modeling Language (VRML) file format, the resulting file was integrated into the MATLAB™ environment. Several functions are now available for accessing the individual joints of the virtual prototype, allowing for the control of individual fingers. Because of this flexibility, the prosthesis can be used to simulate all hand movements recognized by the classification method. Fig. 11 provides screenshots of the virtual prosthesis displaying the five different hand motions. After instantiating the virtual prosthesis in MATLAB™, the position of the individual phalanges can be changed by entering the name and specifying the angle of the joint. Through combining simultaneous movements of several fingers, different hand-positions can be adopted. Listing 1.1 presents the code to set the position of the individual phalanges to assume a fist position. After setting the various joint angles necessary for simulating the desired hand-position, it is possible to assign a label to the position and save it. Several positions can be stored in a file for later reference. This way, the behavior of various prostheses can be captured and sets for various hand positions stored for quick retrieval. Lines 18 to 20 in Listing 1.1 show the process of accessing individual hand positions stored in a file.

Fig. 2 shows the 3D visualization of the five hand movements used for training the decision tree and SVM classifiers described in this contribution. The output of the classifier serves as control input to assign the desired hand-position to the virtual prosthesis model after a movement change from the initial resting hand position is detected.

2.4. MATLAB™ Movement Classification Toolbox

In order to support and accelerate the decision process for the selection of feature-extraction- and classification-methods, a toolbox for MATLAB has been developed. The aforementioned recordings of hand movements can automatically be subjected to feature extraction, classifier training and classifier validation. Both EMG as well as NIR sensor signals are supported with their corresponding features. Various parameters can be set for the individual steps in the classification process. Fig. 13 shows the main toolbox window containing three tabs. The selected first tab has options and dropdown boxes to choose and calculate the desired features from sensor signals. In the selection process, an arbitrary number of sensors as well as feature combinations can be chosen. Furthermore, it is possible to set the window size for the EMG and NIR features.

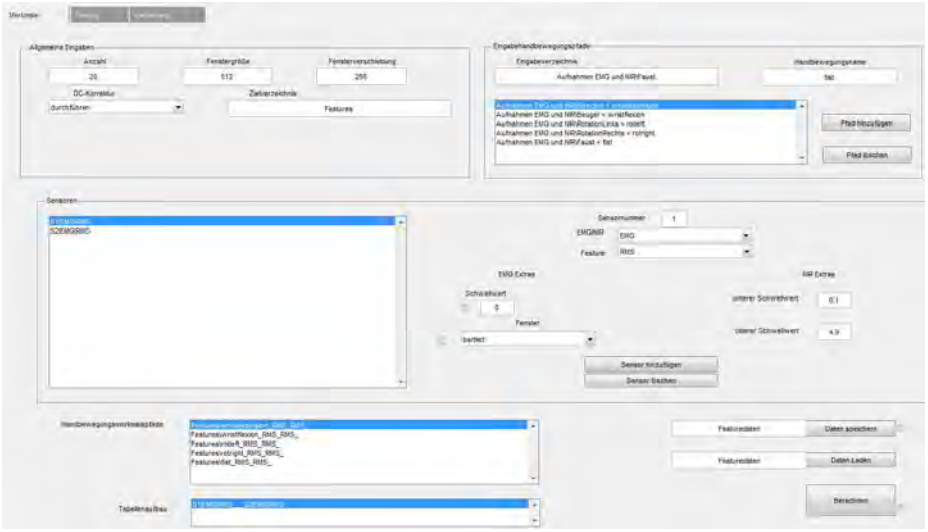


Fig. 13. Main window of the MATLAB toolbox for feature extraction, classifier training and validation.

The program equally offers a high number of parameters for selecting and training the implemented classifiers. The training- and the validation-mode allow to set a threshold to remove noise before feeding the data into a classifier. Furthermore, it is possible to plot selections of training data as well as classification maps. After validation, a detailed report about classification results for individual hand movements can be viewed and saved within MATLAB™. It is also possible to store the parameters used for feature calculation and training model generation. With this combined functionality, feature sets as well as different classifiers can quickly be compared, helping to choose methods and parameters for prosthesis control schemes. The program code has been mod-

ularized as far as possible to offer easy integration of new features as well as classifiers. If novel sensor systems become available, the toolbox can be extended to accommodate for new signal source types.

3. Results

This contribution discloses the modeling, validation and visualization of classification-based prosthesis control schemes. As an example for the individual steps necessary during the classification process, five different hand movements were distinguished using decision tree and SVM classifiers. After feature extraction and training set creation, the trained classifier was validated using existing MES and NIR recordings. The impact of feature and classifier selection is shown with four SVM and decision tree classifiers based on two different feature sets. The classification results were further improved by adding the NIR data from combined EMG-NIR sensors. In addition to the classification process, the behavior of a hand prosthesis is demonstrated through the control of a 3D visualization in MATLAB™ version 7.12.0. As a result, the entire process from training to functional validation and visualization can be seamlessly modeled in one application. Due to the considerable amount of feature extraction as well as classification methods, significant differences in classification accuracy mandate further research focusing on a systematic comparison of feature extraction and classification methods. Research efforts at our department so far resulted in the development of a toolbox for MATLAB™ which enables researchers to select, compare and adapt feature-extraction and classification methods. The current version of the toolbox supports several classification methods including decision trees and support vector machines as well as the extraction of various features from both EMG and NIR signals. Future editions will comprise additional feature calculation and classification algorithms. At the moment, only a limited amount of feature algorithms is available for NIR sensor data. Future research will focus on devising new NIR feature calculation methods. Furthermore, initial digital filtering of raw sensor data to remove noise and artifacts before feature extraction is introduced to increase classification results. Besides improvement of sensor signal processing, current and future research targets the extension of sensor capabilities. For example, the NIR sensor allows changing the area of observation by adjusting the distance between the LED and the photo resistor.

Acknowledgments. The authors thank Dr. Stefan Herrmann and Andrej Gehl for the design and the development of the 3D prosthesis for MATLAB™. Furthermore, we are grateful to Manuel Rosenau for creating a first collection of EMG recordings for testing the classifiers. Finally, we thank Marcus Eckert for his effort towards developing the MATLAB™ movement classification toolbox.

References

1. Arvetti, M., Gini, G., Folgheraiter, M.: Classification of EMG signals through wavelet analysis and neural networks for controlling an active hand prosthesis. In: Proc. IEEE 10th International Conference on Rehabilitation Robotics (ICORR 2007). pp. 531–536 (Jun 2007)
2. Buchenrieder, K.: Dimensionality Reduction for the Control of Powered Upper Limb Prostheses. In: Proc. 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07). pp. 327–333 (Mar 2007)
3. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), software available at http://www.csie.ntu.edu.tw/%7Ecjlin/libsvm
4. Englehart, K., Hudgins, B., Parker, P., Stevenson, M.: Classification of the myoelectric signal using time-frequency based representations. Medical Engineering & Physics 21(6-7), 431–438 (1999)
5. Gehl, A.: Modellierung einer Prothesenhand mit Matlab. Bachelor Thesis, Universität der Bundeswehr München, Neubiberg, Germany (Dec 2010)
6. Herrmann, S., Attenberger, A., Buchenrieder, K.: Prostheses Control with Combined Near-Infrared and Myoelectric Signals. In: EUROCAST 2011, Part II. LNCS, vol. 6928, pp. 602–609 (2011)
7. Herrmann, S., Buchenrieder, K.: Fusion of Myoelectric and Near-Infrared Signals for Prostheses Control. In: Proc. 4th International Convention on Rehabilitation Engineering & Assistive Technology iCREATe'10. pp. 54:1–54:4. Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre, Kaki Bukit TechPark II, Singapore (2010)
8. Izenman, A.J.: Modern Multivariate Statistical Techniques. Springer (2008)
9. León, M., Leija, L., Muñoz, R.: System for the Identification of Multiple Movements of the Hand. In: Proc. 3rd International Conference on Electrical and Electronics Engineering 2006. pp. 1–3 (Sep 2006)
10. Liu, Z., Luo, Z.: Hand Motion Pattern Classifier Based on EMG Using Wavelet Packet Transform and LVQ Neural Networks. In: Proc. IEEE International Symposium on IT in Medicine and Education (ITME 2008). pp. 28–32 (Dec 2008)
11. Peerdeman, B., Boere, D., Witteveen, H.J.B., Huis in 't Veld, M.H.A., Hermens, H.J., Stramigioli, S., Rietman, J.S., Veltink, P.H., Misra, S.: Myoelectric forearm prostheses: State of the art from a user-centered perspective. Journal of rehabilitation research and development 48(6), 719–738 (Aug 2011)
12. Phinyomark, A., Limsakul, C., Phukpattaranont, P.: A Novel Feature Extraction for Robust EMG Pattern Recognition. Journal of Computing 1(1), 71–80 (Dec 2009)
13. Reiter, R.: Eine neue Elektrokunsthand. Grenzgebiete der Medizin 1(4), 133–135 (1948)
14. Theodoridis, S., Koutroumbas, K.: Pattern Recognition. Academic Press, fourth edn. (Aug 2008)

Andreas Attenberger is a PhD student at the Institut für Technische Informatik, Universität der Bundeswehr München under the supervision of Prof. Klaus Buchenrieder. He received his diploma in media informatics from the Ludwig-Maximilians-Universitat Munchen in 2009. The focus of his research

lies on the improvement of natural control schemes for upper limb protheses. His interests include signal acquisition, processing and classification.

Klaus Buchenrieder is a full professor of Informatics at the Universität der Bundeswehr München, Germany. His research and teaching is in the field of embedded systems, reconfigurable system design and design automation. He received a Ph.D. and a M.S. degree in Computer Science from The Ohio State University in Columbus, Ohio. After graduation, he joined the Corporate Research laboratories of Siemens AG in Munich and later the Design Automation Department of Infineon Technologies AG. He holds numerous patents and is a honorary professor of the University of Tubingen, adjunct professor of the Computer and Electrical Engineering department at the University of Arizona and a professor of the Sino-German College of the Tongji University in Shanghai. He is also the founding chair of the Codes/Cashe workshop series on Codesign and of the Consyse workshops on conjoint engineering.

Received: June 1, 2012; Accepted: August 30, 2012.

DOI:10.2298/CSIS120519008P

On Task Tree Executor Architectures Based on Intel Parallel Building Blocks

Miroslav Popovic[1], Miodrag Djukic[1], Vladimir Marinkovic[1], and Nikola Vranic[2]

[1] Faculty of Technical Sciences, Trg D. Obradovića 6,
21000 Novi Sad, Serbia
{miroslav.popovic, miodrag.djukic, vladimir.marinkovic}@rt-rk.com
[2] RT-RK Computer Based Systems LLC, 27 Narodnog fronta 23a,
21000 Novi Sad, Serbia
nikola.vranic@rt-rk.com

Abstract. Our aim was to optimize a SOA control system by evolving the architecture of the service component that transforms system models into task trees, which are then executed by the runtime library called the Task Tree Executor, TTE. In the paper we present the two novel TTE architectures that evolved from the previous TTE architecture and introduced finer grained parallelism. The novel architectures execute TTE tasks as more lightweight TBB tasks and Cilk strands rather than the OS threads, which was the case for the previous TTE architecture. The experimental evaluation based on time needed for TTE reliability estimation, by statistical usage tests, shows that these novel TTE architectures are providing the average relative speedup, RS, from 8x to 11x, over the original TTE, on a dual-core machine. Additional experiments made on eight-core machine showed that RS provided by TTE based on TBB scales perfectly, and goes up to 77x.

Keywords: service oriented architecture, architecture evolution, task trees, parallel programming, parallel building blocks.

1. Introduction

Providing proper parallel data processing is one of the greatest challenges to be dealt with when designing software solutions for management of critical infrastructures, such as oil, gas and electricity distribution systems. The main task of these systems is to provide continuous system supervision and control, based on data acquisition and processing, while fulfilling high availability, reliability, and security standards. Additionally, numerous economic, serviceability, and maintainability aspects regarding different operational activities must be addressed as well. Nowadays all these requirements are typically satisfied by a Service Oriented Architecture (SOA)

based system comprising a complex suite of service components, thus guiding the system designer to the set of necessary data and functionalities that need to be simultaneously served [1-2].

One of the most complicated components for design is a service component that provides various calculations using various models of the system, which commonly take a form of a graph or a tree. The examples of such calculations for the electricity distribution system are network topology analysis, load flow calculation, network state estimation, performance indices, etc. The main factors that are complicating the design of this kind of service components are that these nontrivial calculations have to be performed on large-scale graph models and near to real-time. Designers are also frequently facing the additional economic limitation that they have to somehow reuse legacy software, because of its enormous size – typically millions of lines of FORTRAN code that were developed over a couple of decades.

In our previous work we have used two approaches to design and develop such service components for the electricity distribution systems. The first approach [3-4] is based on: (i) transforming network models into task trees that are managed by Task Tree Executor (TTE) and (ii) refactoring legacy code by introducing callback functions that are executed as TTE tasks. The second approach [5] is based on: (i) repackaging legacy code as DLLs (Dynamic Linkable Libraries) and (ii) executing them as parallel applications by Calculation Engine (CE). The advantage of the second approach is that it requires less development effort and that it is more robust, but the advantage of the first approach is that it provides more parallelism, because it is finer grained than the second approach. TTE runs TTE tasks as separate threads, whereas CE launches the application DLLs within separate processes.

The goal of the work presented in this paper was to evolve the TTE architecture based on threads into the two new TTE architectures based on Intel Parallel Building Blocks (PBB) in order to provide even finer grained parallelism. The first novel TTE architecture is based on Intel Treading Building Blocks (TBB), whereas the second novel TTE architecture is based on Intel Cilk Plus (Cilk). Essentially, the TTE tasks that were executed as threads by the previous TTE [3-4] are now executed as more lightweight TBB tasks by the TTE based on TBB, or as Cilk strands by the TTE based on Cilk.

The advantages of this TTE architecture evolution are threefold. Firstly, both TBB and Cilk are known of being able to provide better multicore CPU utilization than the local OS, such as MS Windows or Linux (i.e. TBB and Cilk can better parallelize their tasks/strands than OS can parallelize its threads). Secondly, both TBB and Cilk provide almost infinite number of tasks, whereas the local OS provides rather limited number of threads within a process (the order of couple of thousands of threads at maximum). Thirdly, the explicit and rather suboptimal CPU load control within the previous TTE architecture is now delegated to the excellent load balancing functionality of the TBB and Cilk runtime libraries within the two novel TTE architectures.

The content of this paper is organized as follows. The related work and the description of target class of software systems that are addressed by the proposed solutions are presented in Subsections 1.1. and 1.2, respectively.

The three TTE architectures are described in Section 2, which is divided into the three subsections. The previous TTE architecture based on OS threads is described in Subsection 2.1, the novel TTE architecture based on TBB is described is Subsection 2.2, and the novel TTE architecture based on Cilk is described is Subsection 2.3. The statistical usage testing method and the results of the experimental evaluation of novel TTE architectures are presented in Sections 3 and 4, respectively. The latter Section 4 is partitioned into the five subsections, which are covering the baseline performance, the performance of the TTE architecture based on TBB, the performance of the TTE architecture based on Cilk, the scalability check for the TTE architecture based on TBB, and the threats to validity of experimental results, respectively. Final conclusions are given in Section 5.

1.1. Related Work

The next two subsections discuss work related to the TTE architectures based on Intel TBB and Intel Cilk Plus, respectively.

Work Related to the TTE Architecture Based on Intel TBB

TBB uses templates for common parallel iteration patterns, enabling programmers to attain increased speed from multiple processor cores without having to be experts in synchronization, load balancing, and cache optimization (see [6]). Generally, TBB provides more comfort to programmers and better results in terms of program speedup when compared to the practice of using raw threads, which was also the case in our particular work presented in this paper.

Two features of TBB that provide the foundation for its robust performance are the TBB work-stealing scheduler and the TBB scalable memory allocator. To prove that, Kukanov and Voss [7] used experiments on several benchmarks to demonstrate the potential scalability of TBB based applications and to show that the TBB allocator is competitive with other allocators.

One of the key advantages of a logical task is that it is much lighter than a thread, e.g. starting and terminating a task on Linux is around 18 times faster than starting and terminating a thread, whereas on Windows, this ratio is more than a 100 (see [8]). Additionally, TBB manages these light units of work very efficiently. Bhattacharjee et al. [9] used real hardware and simulations to detail various scheduler and synchronization overheads in order to assess these overheads on TBB and OpenMP. They found that these can amount to 47% of TBB benchmark runtime and 80% of OpenMP benchmark runtime, i.e. TBB is almost as twice as better when compared to OpenMP in respect to scheduling and synchronization overheads.

Because of all of its features mentioned above, TBB is finding successful applications in various soft real-time systems, sometimes also called near to

real-time systems. One type of such systems is the system managing critical infrastructure, which we are primarily interested in. Another type of such systems is modern video games. Although it might come surprisingly, these two types of systems have much in common. They both use physical models and AI components, they both have real-time requirements, and they both may be classified as complex system.

A significant effort has been made to demonstrate TBB's applicability in modern video games industry. For example, Werth [10] provided useful instructions and hands-on examples on optimizing games architectures with TBB, in his talk on the recent game developer's conference in San Francisco. Several other groups joined this R&D track by trying to create adequate parallel programming frameworks (PPFs) for video games engines.

One notable example in that direction is Cascade, a PPF for video games engine, developed by Tagliasacchi et al. [11]. In Cascade, Cascade tasks are linked by dependencies in a task dependency graph, which is traversed at runtime by the Cascade Job Manager (CJM) that assigns tasks to threads for execution. CJM does this rather efficiently, for example the Cascade implementation of Sequence Alignment algorithm completes 1.5 times quicker than the OpenMP implementation.

As pointed out by the creators of Cascade, TBB is closest in spirit to their system, when compared with other PPFs. However, their claim that TBB does not support explicit construction of task graphs and that graphs are constructed only recursively via spawn call is actually not true. On the other hand Cascade is also very similar to our TTE architecture. The main difference is that Cascade supports acyclic task graphs, whereas TTE supports task trees.

Work Related to the TTE Architecture Based on Intel Cilk Plus

Original Cilk programming language appeared as an extension of C providing language constructs for parallel control and synchronization. This extension was made to be very efficient in terms of runtime overheads, for example the typical cost of spawning an OS thread is 2-6 times the cost of the C function call on a variety of modern processors (see [12]). Once spawned, these parallel threads are scheduled very efficiently on a shared memory multiprocessor (SMP), by the Cilk scheduler that is based on the *work stealing* scheduling method. Blumofe and Leiserson [13] showed that the expected time to execute a well-structured computation on P processors using their work-stealing scheduler is $T_1/P + O(T_\infty)$, where T_1 is the minimum serial execution time of the multithreaded computation and T_∞ is the minimum execution time with an infinite number of processors.

Original Cilk language has been developed, as an ANSI C extension, since 1994 at the MIT. A commercial version of Cilk, called Cilk++, that supports both C and C++, was developed by Cilk Arts, Inc. In 2009, Intel Corporation acquired Cilk Arts, the Cilk++ technology and the trademark. In 2010, Intel released a commercial implementation in its compilers under the name Intel

Cilk Plus. In this paper we use Intel Cilk Plus and refer to it later in the text briefly as Cilk.

Other authors have been successfully using Intel Cilk Plus before us. For example, Kirkegaard and Aleen [14] studed the potential of individual optimizing techniques in terms of speedup. They applied 5 techniques on the Google's AOBench benchmark, to achieve the overall 16.47x speedup.

Similarly, Luk et al. [15] used Intel Cilk Plus to demonstrate their synergetic approach to throughput computing. The experimental results they collected on a dual-socket quad-core Nehalem show that their approach achieves an average speedup of almost 20x over the best serial cases for an important set of computational kernels.

Finally, Agrawal et al. [16] developed the Nabbit, a work-stealing library for execution of task graphs with arbitrary dependencies. They evaluated the performance of Nabbit using a dynamic program representing the Smith-Waterman algorithm. Their results indicate that when task-graph nodes are mapped to reasonably sized blocks, Nabbit exhibits low overhead and scales as well as or better than other scheduling strategies. Interestingly, Nabbit is rather similar both to Cascade and to the TTE architectures presented in this paper. The main difference among them is that Nabbit and Cascade support acyclic task graphs, whereas TTE supports task trees.

1.2. Target Environment

Infrastructure of an industrial control system (ICS) consists of domain specific equipment and smart devices that are connected to Remote Terminal Units (RTUs), which are used for monitoring and control of an industrial process. A modern large scale industrial system typically uses Supervisory Control and Data Acquisition (SCADA) system as a front-end for communication with a network of RTUs.

A separate and independent environment may be laid on top of any SCADA system. This layer, called Intelligent Control System (ICS), encompasses necessary control logic and process related intelligence, which can be very complex. The structure of ICS is shown in Fig. 1. Seen from the SOA standpoint, system exposes the Master Data Service (MDS) that stores logically consistent dataset describing existing elements of the system infrastructure. Besides MDS, other services used for enterprise integrations may be exposed, depending on ICS's role and purpose.

Each of the service components (SCs) in ICS is managing exactly one aspect of the overall system functionality, such as dynamic data management, performing necessary calculations, providing graphical representation of infrastructure elements, etc. The SC providing TTE driven parallel calculations on a given Operational Model (OM) is in the focus of this paper (it is labeled as C-TTE in Fig. 1). OM is a dataset that stores the model of the system. This dataset must be correct, e.g. if ICS manages an electricity distribution system, at least it has to satisfy the first and the second Kirchhoff law.

Interaction between the system user and the services is provided through a thin client application. Usually, there is no demanding data processing inside the client application itself, its only responsibility is to obtain data from particular service in the system periodically, or on user demand.

When the system evolvement aspects are taken into the consideration, one of the most important design goals is to provide the plug-and-play like integration capabilities. Therefore, the main internal communication backbone is designed in accordance with the publisher/subscriber paradigm, which provides loose coupling between service components and services.

For synchronous, point-to-point calls (represented with dashed arrows in Fig. 1), interfaces are provided to allow data access by other services and by external UI clients. However, the communication within the system is predominantly asynchronous, based on publishing and subscribing to different message topics (represented with full arrows in Fig. 1). An important aspect regarding the communication in the system is the fact that all the datasets describe the current infrastructural state of the system, which changes over time.

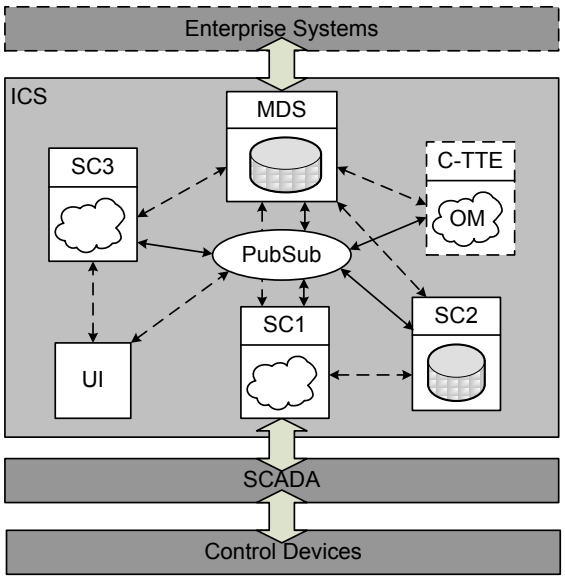


Fig. 1. Target Environment

2. The Three TTE Architectures

The next three subsections present the original TTE architecture based on OS threads, the TTE architecture based on Intel TBB [17], and the TTE architecture based on Intel Cilk Plus [18].

2.1. TTE Architecture Based on Threads

As shown in Fig. 2.a, an application that was refactored from legacy software to operate on slices of system model, which correspond to individual TTE tasks, does that by making use of the TTE application programming interface (API). The TTE API provides the following functions [3]:

1. TS_CreateTaskGraph
2. TS_AddTask
3. TS_DeleteTask
4. TS_SetBottomUpProcFun
5. TS_SetTopDownProcFun
6. TS_ExecuteBottomUp
7. TS_ExecuteTopDown
8. TS_DestroyTaskGraph
9. TS_ExecuteBottomUpSequentially
10. TS_ExecuteTopDownSequentially

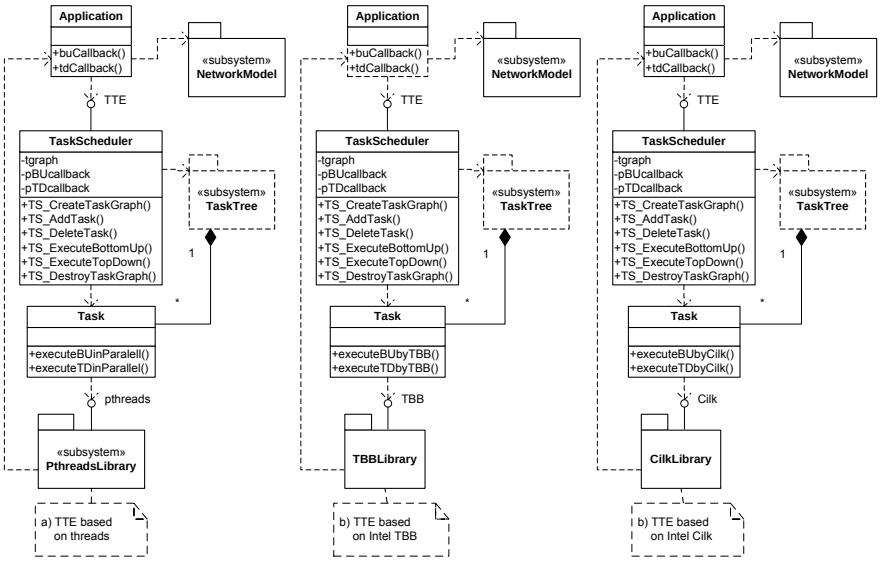


Fig. 2. The three TTE Architectures: (a) Thread-based, (b) TBB-based, (c) Cilk-based

The API function *TS_CreateTaskGraph* creates the task graph; its parameters are the identification (ID) of the root task, the pointer to the bottom-up processing function, the pointer to the top-down processing function, and the maximal number of local OS threads (pthreads) that will be used to execute the task graph in parallel. The bottom-up processing function and top-down processing function are the callback functions, which have the task ID as their parameter. The API function *TS_AddTask* adds a new task to the task graph, given the ID of the predecessor task and the ID of the new

task. The API function *TS_DeleteTask* deletes the given task and all of its successors from the task graph.

The API function *TS_SetBottomUpProcFun* redefines the pointer to the bottom-up processing function, whereas the API function *TS_SetTopDownProcFun* redefines the pointer to the top-down processing function. The API function *TS_ExecuteBottomUp* executes the task graph bottom-up in parallel, whereas the API function *TS_ExecuteTopDown* executes the task graph top-down in parallel. Finally, the API function *TS_DestroyTaskGraph* deletes the task graph.

The last two API functions are used only for the debugging and benchmarking purposes. The first one of them executes the task graph bottom-up sequentially, whereas the second one executes the task graph top-down sequentially. The most frequently used functions are the functions no. 1, 2, 6, 7, and 8. The function no. 3 is used to change the existing task graph, while the functions no. 4 and 5 are used only to redefine the callback functions.

The TTE architecture based on threads comprises two main components, namely the C module *TaskScheduler* (shown as a class in Fig. 2 and Fig.3 for the sake of standard UML representation) and the class *Task*. The module *TaskScheduler* provides the TTE API by exporting its public functions, as listed and discussed above. Internally, this module hides the pointer to the task tree root and the pointers to the callback functions as its private (static) data. As a reaction to external application calls to the functions *TS_CreateTaskGraph*, *TS_AddTask*, *TS_DeleteTask*, and *TS_DestroyTaskGraph*, the module *TaskScheduler* builds and maintains the task tree by adding and deleting instances of the class *Task*.

The class *Task* provides two field members that enable building task trees. These are the pointer to the predecessor task and the list of the successor tasks in the task tree. The function *TS_AddTask* adds a new task by (i) locating its predecessor task, (ii) setting the new task's predecessor field to the address of the predecessor task, and (iii) adding the address of the newly created task to the list of the successor tasks in the corresponding field member of the predecessor task. Deleting a task from the task tree is more complex because deleting a given task means deleting itself and all its successors, and the successors of the successors, i.e. it means deleting the complete sub-tree from the given task and below it.

When it comes to parallel task tree execution, the API function *TS_ExecuteTopDown* starts task tree top-down execution by calling the class *Task* member function *executeTDinParallel* on the root task, which in turn recursively traverses the task tree from its top, i.e. root task, downwards across all the successors, until it reaches all the task tree leafs. In each recursion, this function first calls the top-down callback function and then it starts new local OS threads for each of the current task's successors by calling the *PthreadsLibrary* function *CreateThread* (using the pthreads API). The simplified pseudo code of the class *Task* member function *executeTDinParallel* is the following:

```
executeTDinParallel(task) =
   callback tdCallback(task.id)
   for each successor in task.successors
      CreateThread(executeTDinParallel, successor)
   WaitForAllChildTherads()
```

Similarly, the API function *TS_ExecuteBottomUp* starts task tree bottom-up execution by calling the class *Task* member function *executeBUinParallel* on the root task, which in turn recursively traverses the task tree from its top, i.e. root task, downwards across all the successors until it reaches all the task tree leafs. In each recursion, this function first starts new local OS threads for each of the current task's successors by calling the *PthreadsLibrary* function *CreateThread* (over the pthreads API) and then it calls the bottom-up callback function. The simplified pseudo code of the class *Task* member function *executeBUinParallel* is the following:

```
executeBUinParallel(task) =
   for each successor in task.successors
      CreateThread(executeBUinParallel, successor)
   WaitForAllChildTherads()
   callback buCallback(task.id)
```

2.2. TTE Architecture Based on Intel TBB

The simplified architecture of the complete system that is based on Intel TBB is shown in Fig. 2.b. As shown in Fig. 2.b, the novel TTE architecture based on Intel TBB is almost the same as the previous TTE architecture based on threads. The main difference between these two architectures at the high-level architectural view used in Fig. 2 is that the novel TTE architecture makes use of the Intel TBB runtime library rather than using the local OS (MS Windows or Linux) pthreads library, as was the case in the previous architecture. This evolutionary step was essentially made by modifying the class *Task* such that the member functions *executeTDinParallel* and *executeBUinParallel*, which were responsible for the parallel task tree execution, in the novel architecture delegate parallel top-down and bottom-up task tree execution to new member functions *executeTDbyTBB* and *executeBUbyTBB*, respectively.

This modification was completely transparent to the module *TaskScheduler*, thus the way it builds and maintains the task tree remained unchanged, as well as the way it starts parallel top-down and bottom-up task tree execution. Moreover, and even more importantly, this modification within the TTE architecture was completely transparent to the legacy applications. This was of utmost importance, because legacy applications are so huge in size, they may literally comprise millions of lines of code.

Although, at the high-level of abstraction, simplified pseudo code for the member functions *executeTDinParallel* and *executeBUinParallel* from the previous architecture remains valid for new member functions *executeTDbyTBB* and *executeBUbyTBB*, respectively, implementing them in C++ naturally required using TBB design patterns. More precisely, since TBB tasks are created as instances of C++ classes extending the TBB library class *task*, two auxiliary classes were introduced, namely the class *TbbTaskTD* and the class *TbbTaskBU*. The former is used by the member function *executeTDbyTBB*, whereas the latter is used by the member function *executeBUbyTBB*.

Both of these auxiliary classes are rather simple. Since each TTE task within a TTE task tree is assigned a TBB task, both of these auxiliary classes have a field member that stores the corresponding TTE task. These field members are normally set by the class constructors. The *execute* methods of both auxiliary classes simply call the corresponding new *Task* member function, in particular the *TbbTaskTD* member function calls the *Task* member function *executeTDbyTBB*, whereas the *TbbTaskBU* member function calls the *Task* member function *executeBUbyTBB*.

Once these auxiliary classes were introduced, synthesizing new *Task* member functions was rather straightforward. Concretely, the simplified pseudo code of the function *executeTDbyTBB* is the following:

```
executeTDbyTBB(task) =
   callback tdCallback(task.id)
   if task.successors == Ø  return
   et = TbbTaskTD(null)
   et.set_ref_count(1)
   for each successor in task.successors
      et.increment_ref_count()
      et.spawn( new TbbTaskTD (successor) )
   et.wait_for_all()
   task::destroy(et)
```

In the pseudo code above the name *et* stands for the *empty task*. Similarly, the simplified pseudo code of the function *executeBUbyTBB* is completely symmetrical:

```
executeBUbyTBB(task) =
   if task.successors != Ø
      et = TbbTaskBU(null)
      et.set_ref_count(1)
      for each successor in task.successors
         et.increment_ref_count()
         et.spawn( new TbbTaskBU (successor) )
      et.wait_for_all()
      task::destroy(et)
   callback buCallback(task.id)
```

2.3. TTE Architecture Based on Intel Cilk Plus

The simplified architecture of the system based on Intel Cilk Plus is shown in Fig. 2.c. As shown in Fig. 2.c, the TTE architecture based on Intel Cilk Plus is very similar to the previous two TTE architectures, which were presented in the previous two subsections. The main difference between the TTE architecture based on Intel Cilk Plus and the original TTE architecture based on OS threads is that the former uses the Intel Cilk Plus runtime library, whereas the latter uses the local OS pthreads library.

This evolutionary step is like in Subsection 2.2 made by modifying the class *Task* such that the member functions *executeTDinParallel* and *executeBUinParallel*, which were originally responsible for the parallel task tree execution, now simply delegate parallel top-down and bottom-up task tree execution to new member functions *executeTDbyCilk* and *executeBUbyCilk*, respectively. As such, this modification is again transparent to the module *TaskScheduler*, as well as to all the legacy applications.

Thanks to Cilk's expressiveness, the simplified pseudo code for the member functions *executeTDinParallel* and *executeBUinParallel* from the previous architecture, almost directly map to the pseudo code for new member functions *executeTDbyCilk* and *executeBUbyCilk*, respectively. Essentially, the call to the function *CreateThread* is replaced with the keyword **cilk_for** and the call to the function *WaitForAllChildTherads* is replaced with the keyword **cilk_sync**.

Once these mappings were introduced, synthesizing new *Task* member functions was rather straightforward. Consequently, the pseudo code of the function *executeTDbyCilk* is the following:

```
executeTDbyCilk(task) =
   callback tdCallback(task.id)
   for each scsr in task.successors
      cilk_spawn scsr.executeTDbyCilk(scsr)
   cilk_sync
```

In the pseudo code above the name *scsr* stands for the *successor task*. Similarly, the pseudo code of the function *executeBUbyCilk* is completely symmetrical:

```
executeBUbyCilk(task) =
   for each scsr in task.successors
      cilk_spawn scsr.executeBUbyCilk(scsr)
   cilk_sync
   callback buCallback(task.id)
```

3. Statistical Usage Testing

We used the method published in [19] for statistical usage testing and operational reliability estimation of all three TTE architectures. The method is mostly based on the approach created by D.M. Woit [20-23] and on the following work of several authors [24-27] that modernized that approach and adapted it to a form of the model-based testing.

For the sake of completeness of this paper, we provide a brief overview of the method [19] in this section. We start with some definitions, then provide formulas for the number of test cases *N* and for the confidence level *M*, and finally outline the method in a form of a series of steps.

A *task* τ is a callback function that executes as a local OS thread. A *task tree* is an undirected radial (i.e. acyclic) graph of tasks *TG* whose nodes are tasks interconnected with links indicating predecessor-successor relations. A task tree comprises a set of *k* tasks $TK = \{\tau_1, \tau_2, \ldots, \tau_k\}$, and a set of (*k*-1) links $L = \{l_1, l_2, \ldots, l_{(k-1)}\}$.

A *task tree execution path*, a.k.a. a *path* in a task tree or a *trace*, is a sequence of terminations of individual tasks $\tau_1\tau_2\ldots\tau_k$ during the task tree execution. The length of this sequence is always equal to k. A *task forest* is a series of task trees of the same complexity (the same number of nodes) that is generated as a test suite. A *test case* is a single task tree execution described by the corresponding path.

Let r_t be a software product *tree-reliability* and r_p be a *path-reliability*. Then it may be easily shown that the product reliability *r* is obtained by multiplying the two:

$$r = r_t r_p \,. \tag{1}$$

If we further assume that $r_t = r_p$, then:

$$r_t = r_p = r^{1/2} \,. \tag{2}$$

Similarly, let M_t be a *tree-confidence-level* and M_p be a *path-confidence level*. Then it may be easily shown that the total confidence level *M* is the sum of the two:

$$M = M_t + M_p \,. \tag{3}$$

If we further assume $M_t = M_p$, then:

$$M_t = M_p = M/2 \,. \tag{4}$$

Therefore, when given *r* and *M* we calculate the requested number of trees N_t and number of paths N_p for each tree as:

$$N_t = N_p = \log_r^{1/2} (M/2) \,. \tag{5}$$

Finally, the total number of test cases *N* is obtained as a simple product of N_t and N_p:

$$N = N_t N_p = (\log_r^{1/2} (M/2))^2 \,. \tag{6}$$

This means that we simply have to generate N_t task trees and execute them N_p times each. Thus the method of statistical testing and reliability estimation for applications based on task trees consists of the following steps:

1. Given the desired level of product reliability, calculate N_t and N_p.
2. Generate N_t task trees.
3. Execute each task tree N_p times.
4. Check the coverage metrics report.
5. If the report shows poor coverage, return to step 2.
6. Report any unexpected behavior to the design and implementation team.

4. Experimental Evaluation

Firstly, Statistical Usage Testing (SUT) and reliability estimation method described in the previous section was used to test all the TTE architectures. Secondly, SUT was used to evaluate the performance of the two new TTE architectures based on TBB and Cilk (see subsections 4.2 and 4.3) with respect to the original TTE architecture based on OS threads, which served as a baseline (see section 4.1).

The measure of the performance that was used in the experiments was the time in seconds that was needed to execute all the N test cases from the given test suite. For the sake of completeness of the paper we provide the execution time measurements data for individual test suits for both TTE architectures, and for the sake of easier performance comparison between the two architectures we provide the relative speedup (*RS*) calculation results. The relative speedup *RS* is defined as the ratio:

$$RS = T_p / T_n \,. \tag{7}$$

where T_p is the test suite execution time for the TTE architecture based on threads and T_n is the test suite execution time for the TTE architecture based on TBB (in subsection 4.2) or on Cilk (subsections 4.3).

All the SUT based measurements were conducted on the dual-core symmetric multiprocessor, Intel® Core(TM) i5 CPU M 520 @ 2.4 GHz, 4 GB RAM, with Windows7 Professional® 64-bit OS.

After conducting SUT based measurements, we made an additional scalability check for the TTE architecture based on TBB on the Intel Server Board SE8501HW4 with 4 Xeon MP Dual Core CPU, facilitating the total of 8 cores operating on 2.4 GHz, with 12 GB of main memory. Software used in the experiments is OS CentOS 5.4 and open Intel TBB (see subsection 4.4). Unfortunately, open Intel Cilk Plus was still not mature enough and its port on CentOS 5.4 was not available at the time of this writings, so we were not able to make the same check for TTE architecture based on Cilk.

At the end of this section we discuss various threats to validity of our experimental results (see subsection 4.5).

4.1. Baseline: Performance of the TTE Based on OS Threads

Table 1 provides T_p values and test verdicts. The columns of Table 1 are organized as follows. The column "No Tasks" contains the number of TTE tasks used to construct task trees, the column "No Trees" shows the number of tasks that may be constructed by the given number of TTE tasks, the next three columns within the common column "Duration [s]" show test suites execution time in seconds for the three distinctive values of desired reliability *r* (*r*=0.9, *r*=0.95, and *r*=0.99), and the last column "Verdict" contains the test verdict.

As could be seen from Table 1, test suite execution time increases with the number of TTE tasks and with the value of desired reliability *r*. As the last column indicates, TTE based on threads successfully passed all the tests.

Table 1. Measurements for TTE Based on Threads

No Tasks	No Trees	Duration [s] *r*=0.9	*r*=0.95	*r*=0.99	Verdict
1	1	0	1	11	Pass
2	1	2	7	196	Pass
3	2	2	10	273	Pass
4	6	4	14	298	Pass
5	24	4	15	365	Pass
6	120	5	17	426	Pass
7	720	5	20	485	Pass
8	5040	18	35	558	Pass

4.2. Performance of the TTE Based on Intel TBB

The measured data and the calculated results are given in the following two tables below. Table 2 provides T_n values and test verdicts, whereas Table 3 provides calculated *RS* values.

The columns of Table 2 are organized in the same way as the columns of Table 1. Similarly, as in Table 1, test suite execution time increases with both number of tasks and the value of given reliability. The latter, again, causes faster growth of the test suite execution time than the former. The last column of Table 2 shows that TTE based on TBB also passed all the tests successfully. All of this seems very similar, but the measured values of test suites execution times are drastically different. Obviously, it took much less time for TTE based on TBB to complete all the tests than it did for the TTE based on threads. This is even more evident from Table 3.

At this point, it seems appropriate to mention that we were not able to calculate some of the values of relative speedup *RS* from the raw data in

Tables 1 and 2, because some of the values of test suite execution times were 0. Therefore, the corresponding values of *RS* were undefined (dividing 0 with 0 is undefined, and dividing the nonzero number with 0 converges towards infinity, which does not reflect reality in terms of realistic speedup that could be achieved). On the other hand, the test suites execution times are realistically always greater than zero – zero value is only a consequence of imprecise measurements. Finally, since test suites execution times were going up to several hundreds of seconds for the TTE architecture based on threads, we rounded all the 0 second measurements, in Tables 1 and 2, to the 1 second values. By doing so, we introduced a small error, which may be neglected, but we were able to provide *RS* values presented in Table 3.

Table 2. Measurements for TTE Based on TBB

No Tasks	No Trees	Duration [s] *r*=0.9	*r*=0.95	*r*=0.99	Verdict
1	1	0	0	7	Pass
2	1	1	1	20	Pass
3	2	0	1	23	Pass
4	6	0	1	27	Pass
5	24	0	1	30	Pass
6	120	1	2	34	Pass
7	720	0	1	37	Pass
8	5040	8	8	49	Pass

Table 3. Calculated Values of Relative Speedup *RS* for TTE Based on TBB

No Tasks	No Trees	Relative Speedup RS *r*=0.9	*r*=0.95	*r*=0.99	Average over forests
1	1	1.00	1.00	1.57	1.19
2	1	2.00	7.00	9.80	6.27
3	2	2.00	10.00	11.87	7.96
4	6	4.00	14.00	11.04	9.68
5	24	4.00	15.00	12.17	10.39
6	120	5.00	8.50	12.53	8.68
7	720	5.00	20.00	13.11	12.70
8	5040	2.25	4.38	11.39	6.00
Average over *r*		3.16	9.98	10.43	7.86

Table 3 shows the values of relative speedup *RS* of test suite execution on new and previous TTE architectures, for various numbers of tasks and desired operational reliability *r* figures. The columns of Table 3 are organized similarly as the columns of Tables 1 and 2. The additional row shows the average *RS* calculated over different values of desired operational reliability *r*, whereas the last column shows the average *RS* evaluated over a different

number of tasks (rather than the test suit verdict like in Tables 1 and 2). The bottom-right cell of Table 3 shows an overall *RS* average when evaluated over all *RS* values.

As expected, the relative speedup *RS* increased both with the number of tasks for a given operational reliability *r*, and with the desired operational reliability *r* for a given number of tasks. Obviously, RS grows much faster with the desired *r* than with the number of tasks, which appears quite natural, because the needed testing effort increases much more with operational reliability *r* than with the number of tasks. As a consequence of these trends, both the average *RS*, calculated per task, increases with the number of tasks, and the average *RS*, calculated per given operational reliability *r*, increases with the value of *r*.

The overall average relative speedup is 7.86 (bottom-right cell in Table 3), which is quite a good result for the dual-core target machine we used in the experiments. Of course, it would be interesting to see how this average speedup of around 8 changes with the number of available cores in the target platform. In Subsection 4.4 we conduct more experiments in that direction in order to check the scalability of the proposed solution.

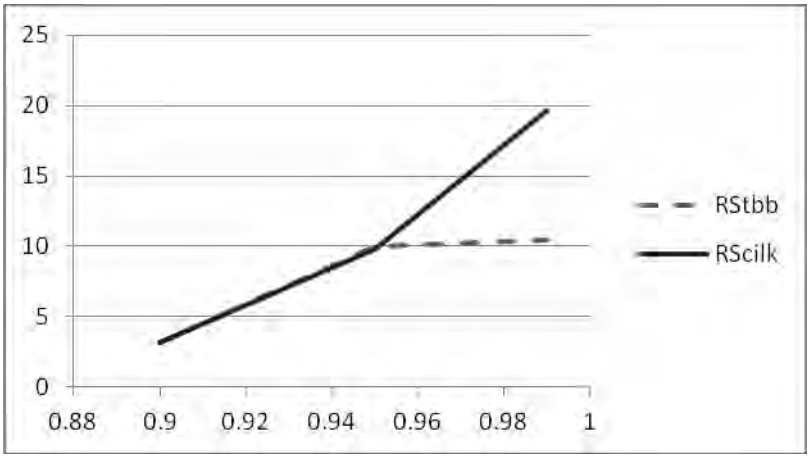


Fig. 3. The average relative speedup *RS* as a function of reliability *r*. The dashed curve shows the average RS values for the TTE based on TBB, whereas the full curve shows the average RS values for the TTE based on Cilk

Another important fact that may be seen by looking at the values of average *RS* in the last row of Table 3, is that average *RS* is around 3 only for the value *r*=0.9. For the values of *r* that are greater than 0.9, average *RS* is around 10, so the test suite execution on the novel TTE architecture is an order of magnitude faster than on the previous architectures, for the greater values of *r* (0.95 and 0.99 in Table 3). This fact becomes even more obvious by observing Fig. 3, which shows the average relative speedup *RS* as a function of a given operational reliability *r* (see the curve *RStbb* in Fig. 3).

4.3. Performance of the TTE Based on Intel Cilk Plus

The measured data and the calculated results are given in the following two tables below. Table 4 provides T_n values and test verdicts, whereas Table 5 provides calculated *RS* values.

Table 4. Measurements for TTE Based on Cilk

No Tasks	No Trees	Duration [s] *r*=0.9	*r*=0.95	*r*=0.99	Verdict
1	1	1	1	9	Pass
2	1	1	1	10	Pass
3	2	1	1	12	Pass
4	6	1	1	13	Pass
5	24	1	1	16	Pass
6	120	1	1	17	Pass
7	720	1	2	20	Pass
8	5040	7	8	30	Pass

Table 5. Calculated Values of Relative Speedup *RS* for TTE Based on Cilk

No Tasks	No Trees	Relative Speedup RS *r*=0.9	*r*=0.95	*r*=0.99	Average over forests
1	1	1.00	1.00	1.22	1.07
2	1	2.00	7.00	19.60	9.53
3	2	2.00	10.00	22.75	11.58
4	6	4.00	14.00	22.92	13.64
5	24	4.00	15.00	22.81	13.94
6	120	5.00	17.00	25.06	15.69
7	720	5.00	10.00	24.25	13.08
8	5040	2.57	4.38	18.60	8.51
Average over *r*		3.20	9.80	19.65	10.88

The columns of Table 4 are organized in the same way as the columns of Table 2. Similarly, as in Table 2, test suite execution time increases with both number of tasks and the value of given reliability *r*. Again, the latter causes faster growth of the test suite execution time than the former. The last column of Table 4 shows that TTE based on Cilk successfully passed all the tests. The measured values of test suites execution times for TTE based on Cilk are even smaller than the corresponding times for the TTE based on OS threads. This fact becomes more evident by observing Table 5.

Table 5 shows the values of relative speedup *RS* of test suite execution on the TTE based on Intel Cilk and on the TTE based on OS threads, for various numbers of tasks and desired operational reliability *r* figures. Table 5 is organized in the same way as Table 3.

The results of the qualitative analysis of data given in Table 5 are practically the same as the previous qualitative analysis of data given in Table 3. Again, both the average *RS*, calculated per task, increases with the number of tasks, and the average *RS*, calculated per given operational reliability *r*, increases with the value of *r*. And again *RS* grows much faster with the desired *r* than with the number of tasks.

The overall average relative speedup is 10.88 (bottom-right cell in Table 5), which is a good result for the dual-core target machine we used in the experiments. Of course, it would be interesting to see how this average speedup of around 11x changes with the number of available cores in the target platform, and we have a plan to conduct more experiments in that direction in the future.

But, even more important fact that may be seen by observing the values of average *RS* in the last row of Table 5, is that overall average *RS* of 11x is actually much limited by the *RS* value of around 3x for r=0.9. For the values of *r* greater than 0.9, average *RS* goes up to 20x (for r=0.99). So after analyzing this data, one becomes aware that the novel TTE architecture provides scalable performance relative to given operational reliability *r*. This fact becomes even more obvious by observing Fig. 3, which illustrates the average relative speedup *RS* as a function of a given operational reliability *r* (see the curve *RScilk* in Fig. 3).

Finally, Fig. 3 makes it possible to compare the two TTE solutions that are based on Intel Parallel Building Blocks. We see from Fig. 3 that the *RS* has greater values for the TTE base on Cilk than for the TTE based on TBB. The values for the former go up to 10x, whereas the values for the latter go up to 20x.

4.4. Scalability Check for the TTE Based on Intel TBB

The results presented in the previous subsections show that performance of newly developed TTE architectures scale rather well with respect to the operational reliability *r*. But, all the previously described experiments were conducted on the dual-core machine and on small task trees consisting of up to 8 tasks. In this subsection we check performance scalability of the TTE based on TBB with respect to the number of processor cores and with respect to the number of tasks in randomly generated large task trees.

For this purpose we conducted the three series of experiments for the three particular numbers of tasks (k) that were used to randomly construct tasks trees, namely k=600, k=800, and k=1000 tasks, respectively. The task trees were randomly generated by the previously developed component *TreeGrower*, which is described in [19]. In each series of experiments we indirectly measured the *RS* of TTE based on TBB in respect to the original TTE based on OS threads for various numbers of processor cores N_c, from N_c=2 to N_c=8 with the step 2 (i.e. N_c=2,4,6,8).

The *RS* was indirect measured as follows. We directly measured the execution times of SUT tests targeting r=0.9 for both TTE based on OS threads and TTE based on TBB, three times each, then we calculated the mean values of execution times, and finally we calculated the corresponding *RS* values. The final results are given in Table 6 and they are illustrated in Fig. 4.

Table 6. Relative Speedup *RS* for various numbers of cores and tasks. RS600 is the *RS* for k=600, RS800 is the *RS* for k=800, and RS1000 is the *RS* for k=1000 tasks

No Cores	RS600	RS800	RS1000
2	29.45	46.17	52.75
4	36.14	57.34	64.49
6	41.45	65.46	72.99
8	43.98	69.39	77.17

Table 6 is organized as follows. The column “No Cores” indicates the number of processor cores that were utilized by TTE based on TBB. The columns “RS600”, “RS800”, and “RS1000” show the RS for k=600, k=800, and k=1000 tasks, respectively.

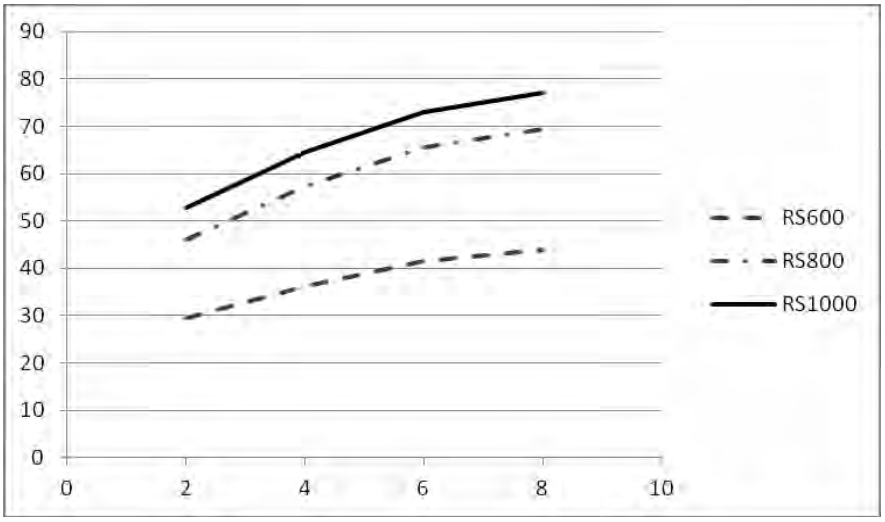


Fig. 4. The average relative speedup *RS* as a function of the number of cores N_c and the number of tasks N_t. The dashed curve shows the *RS* for k=600, the dotted-dashed curve shows the RS for k=800, and the full curve shows the *RS* for k=1000

As indicated by Fig. 4, the performance, in terms of relative speedup *RS* of the TTE based on TBB in respect to TTE based on OS threads, scales perfectly with both the number of processor cores and the number of tasks within a task tree. The *RS* increase linearly with the number of cores and

logarithmically with the number of tasks, and it goes up to 77x for N_c=8 cores and k=1000 tasks.

4.5. Threats to validity of experimental results

At the end of this section we briefly address the threats to validity of the presented results. From all the kinds of threats, the *threats to the external validity* are the most serious threats for the presented results, because repeating the experiments on a different platform (machine and operating system) would very likely yield different results than those shown in Tables 1-6. The only way to address this issue is to repeat the experiments on several different platforms, and that remains to be done in our future work.

The other two kinds of threats, namely the *threats to internal validity* and the *threats to construct validity* do exist, but can be neglected. We minimized the former threats by disconnecting the target machine from the Internet and by closing all the other applications. The latter threats reduce here to imprecision of measuring time, which obviously can be neglected.

5. Conclusions

Application of parallel programming techniques to design of software solutions is a promising trend. In this paper we have shown an approach to apply parallel programming techniques based on Intel Parallel Building Blocks to a class of service components within SOA based industrial systems. Moreover, we have shown an approach to introduce either the Intel TBB library, or Intel Cilk Plus library, instead of the conventional OS threads library through a corresponding evolutionary step with minimal adaptations of the legacy TTE architecture. Such evolutionary approaches to architecting new system versions are necessary because legacy software may be of extreme size, typically measured in millions of lines of code.

The results of the approach are two novel TTE architectures. The first one is based on Intel TBB that executes TTE tasks as TBB tasks, whereas the second one is based on Intel Cilk Plus that executes TTE tasks as Cilk strands. Essentially, novel TTE architectures use finer grained parallelism, which yields better multicore CPU utilization. The first novel TTE architecture based on TBB exhibited the average relative speedup *RS* of around 8x, and the maximal *RS* of 10x, over the original TTE architecture based on pthreads. Similarly, and even better, the second novel TTE architecture based on Cilk achieved the average *RS* of around 11x, and the maximal *RS* of 20x, over the original TTE architecture based on pthreads.

Additional scalability check that was made for the first novel TTE architecture based on TBB showed that its performance in terms of relative speedup *RS* scales perfectly with both the number of processor cores and the number of tasks within a task tree. The *RS* increase linearly with the number

of cores and logarithmically with the number of tasks, and it goes up to 77x for N_c=8 cores and k=1000 tasks.

In our future work we plan (i) to make the scalability check for TTE architecture based on Cilk if and when open Intel Cilk Plus port for CentOS 5.4 becomes available, (ii) to explore other algorithms for parallel task tree execution and their implementations, (iii) to evolve TTE architecture in order to support also other non Intel multicores, as well as heterogeneous multicores, and (iv) to develop a distributed TTE architecture for a system with many heterogeneous multicores.

Acknowledgments. This work has been partly supported by the Serbian Ministry of Education & Science, through grants No. III 44009 and TR 32031.

References

1. Komoda, N., Service Oriented Architecture (SOA) in Industrial Systems. In the Proceedings of IEEE International Conference on Industrial Informatics. IEEE CPS, Los Alamitos, CA, USA, pp. 1-5. (2006)
2. Popovic, I., Vrtunski, V., Popovic, M.: Formal Verification of Distributed Transaction Management in a SOA Based Control System. In Proceedings of the 18th IEEE International Conference and Workshops on Engineering of Computer Based System. IEEE CPS, Los Alamitos, CA, USA, 206-215. (2011)
3. Popovic, M., Basicevic, I., Vrtunski, V.: A Task Tree Executor: New Runtime for Parallelized Legacy Software. In Proceedings of the 16th IEEE International Conference and Workshops on Engineering of Computer Based System. IEEE CPS, Los Alamitos, CA, USA, 41-47. (2009)
4. Basicevic, I., Jovanovic, S., Drapsin, B., Popovic, M., Vrtunski, V.: An Approach to Parallelization of Legacy Software. In Proceedings of the 1st Eastern European Regional Conference on the Engineering of Computer Based Systems. IEEE CPS, Los Alamitos, CA, USA, 42-48. (2009)
5. Trivunovic, B., Popovic, M., Vrtunski, V.: An Application Level Parallelization of Complex Real-Time Software. In Proceedings of the 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems. IEEE CPS, Los Alamitos, CA, USA, 253-257. (2010)
6. Reinders, J.: Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelisim. O'Reilly Media, Inc., 1005 Gravestein Highway North, Sebastopol, CA, USA. (2007)
7. Kukanov, A., Voss, M. J.: The Foundations for Scalable Multi-core Software in Intel Threading Building Blocks. Intel Technology Journal, Vol. 11, No. 4, 309-322. (2007)
8. Popovici, N., Willhalm, T.: Putting Intel® Threading Building Blocks to Work. In Proceedings of the 1st International Workshop on Multicore Software Engineering, ACM Press, New York, NY, USA, 3-4. (2008)
9. Bhattacharjee, A., Contreras, G., and Martonosi, M.: Parallelization Libraries: Characterizing and Reducing Overheads. ACM Transactions on Architecture and Code Optimization, Vol. 8, No. 1, Article 5, 1-29. (2011)
10. Werth, B.: Optimizing Game Architectures with Intel® Threading Building Blocks. Intel Software Network (2009). [Online]. Available: http://software.intel.com/en-

us/articles/optimizing-game-architectures-with-intel-threading-building-blocks/ (current May 2012)
11. Tagliasacchi, A., Dickie, R., Couture-Beil, A., Best, M.J., Fedorova, A., Brownsword, A.: Cascade: A Parallel Programming Framework for Video Game Engines. In Proceedings of the Workshop on Parallel Execution of Sequential Programs on Multi-core Architectures. Institute of Computing Technology, Chinese Academy of Sciences, 47-54. (2008)
12. Randall, K.H.: Cilk: Efficient multithreaded computing. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA. (1998)
13. Blumofe, R.D., Leiserson, C.E.: Scheduling Multithreaded Computations by Work Stealing. Journal of the ACM, Vol. 46, No. 5, 720 –748. (1999)
14. Kirkegaard, K., Aleen, F.: Using Intel® Cilk™ Plus to Achieve Data and Thread Parallelism: A Case Study for Visual Computing, 2011. [Online]. Available: http://software.intel.com/en-us/articles/data-and-thread-parallelism/ (current May 2012)
15. Luk, C. K., Newton, R., Hasenplaugh, W., Hampton, M., Lowney, G.: A Synergetic Approach to Throughput Computing on x86-Based Multicore Desktops. IEEE Software, Vol. 28, No. 1, 39-50. (2011)
16. Agrawal, K., Leiserson, C.E., Sukha, J.: Executing Task Graphs Using Work-Stealing. In the Proceedings of the 24th IEEE International Parallel & Distributed Processing Symposium. IEEE CPS, Los Alamitos, CA, USA, 1-12. (2010)
17. Popovic, M., Djukic, M., Marinkovic, V., Vranic, N.: A Task Tree Executor Architecture Based on Intel Threading Building Blocks. In Proceedings of the 19th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems. IEEE CPS, Los Alamitos, CA, USA, 201-209. (2012)
18. Popovic, M., Basicevic, I.: An Intel Cilk Plus Based Task Tree Executor Architecture. In Proceedings of the11th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems. WSEAS Press, 30-35 (2012)
19. Popovic, M., Basicevic, I.: Test case generation for the task tree type of architecture. Information and Software Technology, Vol. 52, No. 6, 697–706. (2010)
20. Woit, D.M.: Specifying Operational Profiles for Modules. In Proceedings of the 1993 ACM SIGSOFT international symposium on Software testing and analysis. ACM Press, New York, NY, USA, 2-10. (1993)
21. Woit, D.M.: Estimating Software Reliability with Hypothesis Testing. Technical Report CRL-263, McMaster University. (1993)
22. Woit, D.M.: Operational Profile Specification, Test Case Generation, and Reliability Estimation for Modules. Ph.D. Thesis, Queen's University Kingstone, Ontario, Canada. (1994)
23. Woit, D.M.: A Framework for Reliability Estimation. In Proceedings of the 5th IEEE International Symposium on Software Reliability Engineering. IEEE CPS, Los Alamitos, CA, USA, 18-24 (1994)
24. Popovic, M., Velikic, I.: A Generic Model-Based Test Case Generator. In Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems. IEEE CPS, Los Alamitos, CA, USA, 221-228. (2005)
25. Popovic, M., Basicevic, I., Velikic, I., Tatic, J.: A Model-Based Statistical Usage Testing of Communication Protocols. In Proceedings of the 13th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems. IEEE CPS, Los Alamitos, CA, USA, 377-386. (2006)

26. Popovic, M. Kovacevic, J.: A Statistical Approach to Model-Based Robustness Testing. In Proceedings of the 14th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems. IEEE CPS, Los Alamitos, CA, USA, 485-494. (2007)
27. Popovic, M.: Communication Protocol Engineering. CRC Press, Boca Raton, FL, USA. (2006)

Prof. Miroslav Popovic received his M.Sc. and Ph.D. degrees in electrical and computer engineering from the Faculty of Technical Sciences at the University of Novi Sad, Novi Sad, Serbia, in 1984 and 1990, respectively. He started his career as an assistant professor at the Faculty of technical sciences, where he remained working to the present day. He was promoted to a tenured professor in 2002. He is currently the head of the Chair of computer engineering. He wrote the book Communication Protocol Engineering (Boca Raton, Florida, USA: CRC Press, 2006) and about 150 papers published in international and domestic journals and conference proceedings. His current research interests are in the areas of parallel programming, model-based development, testing, and verification. Prof. Popovic is the member of the program committee of the IEEE Annual Conference on Engineering of Computer Based Systems (ECBS), and also the member of IEEE, IEEE Computer Society, IEEE TC on ECBS, and ACM.

Miodrag Djukic graduated from the Faculty of Technical Sciences, University of Novi Sad, in 2007, received M.Sc. one year later from the same university. His research interest is mostly focused on compilers and software tools in general, applications of artificial intelligence, and computer graphics. He is a teaching assistant at the Faculty of Technical Sciences and works on several projects for RT-RK, Research and Development Institute for Computer Based Systems. He is the member of IEEE, IEEE Computer Society, and IEEE TC on ECBS.

Vladimir Marinkovic graduated and received M.Sc. degree in electrical and computer engineering from the Faculty of Technical Sciences, University of Novi Sad, Serbia, in 2009 and 2010 respectively. He is currently pursuing Ph.D. degree from the same university. His research interests are focused on both parallelization of programs for execution on multiprocessors and multi-core processors, and compilers. In the year of 2011, he was elected to the position of teaching assistant at RT-RK, Research and Development Institute for Computer Based Systems. He is scholar of the Ministry of Science and Technology from the school year 2010/2011.

Nikola Vranic received his B.Sc. and M.Sc. degrees in computer engineering and computer communications at the Faculty of technical sciences, University of Novi Sad. He is currently on Ph.D. studies at the same University. He has been working on security of optical communication lines, developing compiler modules for parallelization, digital television etc. His research interests include multicore systems, code parallelization, cryptography, android, Google TV, etc. He is author and coauthor of a dozen of scientific papers in country and aboard. He is currently employed like Google TV software engineering in the company RT-RK LLC and working as assistant at the Faculty of Technical Sciences.

Received: May 19, 2012; Accepted: August 30, 2012.

DOI: 10.2298/CSIS120601009W

Modeling and Verifying the Ariadne Protocol Using Process Algebra

Xi Wu[1], Huibiao Zhu[1], Yongxin Zhao[2], Zheng Wang[3], and Si Liu[4]

[1] Shanghai Key Laboratory of Trustworthy Computing
Software Engineering Institute, East China Normal University
3663 Zhongshan Road (North), Shanghai, China, 200062
{xiwu,hbzhu}@sei.ecnu.edu.cn
[2] School of Computing, National University of Singapore, Singapore
zhaoyx@comp.nus.edu.sg
[3] Beijing Institute of Control Engineering, China
wangzheng@sei.ecnu.edu.cn
[4] Department of Computer Science, University of Illinois at Urbana-Champaign
siliu3@illinois.edu

Abstract. Mobile Ad Hoc Networks (MANETs) are formed dynamically by mobile nodes without the support of prior stationary infrastructures. In such networks, routing protocols, particularly secure ones are always the essential parts. Ariadne, an efficient and well-known on-demand secure protocol of MANETs, mainly concerns about how to prevent a malicious node from compromising the route. In this paper, we apply the method of process algebra Communicating Sequential Processes (CSP) to model and reason about the Ariadne protocol, focusing on the process of its route discovery. In our framework, we consider the communication entities as CSP processes, including the initiator, the intermediate nodes and the target. Moreover, we also propose an intruder model allowing the intruder to learn and deduce much information from the protocol and the environment. Note that the modeling approach is also applicable to other protocols, which are based on the on-demand routing protocols and have the route discovery process. Finally, we use PAT, a model checker for CSP, to verify whether the model caters for the specification and the non-trivial secure properties, e.g. nonexistence of fake path. Three case studies are given and the verification results naturally demonstrate that the fake routing attacks may be present in the Ariadne protocol.

Keywords: Formal Verification, CSP, Mobile Ad Hoc Networks, Ariadne.

1. Introduction

Wireless communication technology [12] has become one of the most promising technologies. Mobile Ad Hoc Networks (MANETs) [22,39] consist of groups of wireless mobile devices (laptops, PDAs, sensors, etc.), being completely self-configuring and self-organizing, and are independent of any existing fixed infrastructure. In such networks, nodes can forward the data packets for each

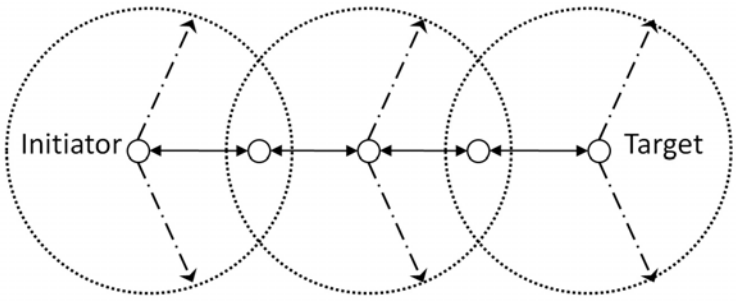


Fig. 1. Communication Model Diagram of Multi-hop Mobile Ad Hoc Network

other through the mutual cooperation. Consequently, even if there is no direct link between two nodes, they also can communicate with each other through the intermediate node multi-hop routing technology, thus widening the range of the data packet transmission. Moreover, nodes can move arbitrarily within, join in, or leave the network dynamically, which makes the whole network quickly and easily set up as needed. Due to these novel features, MANETs have been widely applied in many fields including military, ambient intelligence and emergency contingencies. Figure 1 shows the communication model diagram of multi-hop mobile ad hoc networks.

In such networks, routing protocols [9,10,32,43], particularly secure ones are always the essential factors since they are the major concerns about how to prevent a malicious node from compromising the route. Malicious nodes may cause some typical security issues such as the attacks of denial-of-service and tunneling which redirect the traffic of the networks, the attacks of spoofing that the intruder node may masquerade as the other nodes, and the attack called fabrication of false routing messages. Ariadne [10], as an extension to the dynamic source routing (DSR) protocol [11], proposed by Hu *et al.*, is a new secure on-demand ad hoc network routing protocol for preventing attackers and security vulnerabilities.

Many research efforts have been addressed to analyze and improve the Ariadne protocol. Hu *et al.* evaluated its performance based on simulation [10]. Sivakumar *et al.* proposed some modifications to improve its resiliency [33]. All these works, however, do not investigate the protocol using formal methods and may not take into account the security and correctness. In addition, Lin *et al.* have already found some drawbacks of this protocol, describing them in natural language [13] and Buttyán *et al.* applied a mathematical framework in analyzing the protocol and finding out attacks on it [1,5]. They have done well in analyzing the protocol, if only they had given some verifications. In formal literature, as far as we know, only Pura *et al.* have already modeled the Ariadne protocol using HLPSL and applied AVISPA to validate its security properties [28]. However, they focus more on the use of the tools than the analysis of the protocol itself. Thus, the research for an approach to modeling and verifying the Ariadne

protocol is still challenging. In this paper, we use formal methods to model the protocol and use the process algebra tool PAT to verify whether the achieved model caters for the specification and the non-trivial secure properties.

Lowe *et al.* first applied the method of process algebra Communicating Sequential Processes (CSP) to model and analyze a security protocol, the TMN protocol [19]. CSP is a well-known process algebra in modeling and verifying the reliability, the sequential consistency and the sercurity in concurrent systems and widely used in [18,19,23,29]. Besides, many researchers, such as Zhu and his students, have successfully used CSP to model and analyze the protocols and the web service systems [7,41,42]. Moreover, a lot of automated model checkers for analyzing and understanding systems described by CSP have been produced, such as Process Analysis Toolkit (PAT) [34]. Inspired by Lowe's work, we use CSP to model the Ariadne secure routing protocol. This protocol has two phases: the route discovery and the route maintain. Due to the facts that the route maintain is based on the route discovery and the intrusion tends to occur in the discovery phase, in this paper, we focus on the Ariadne route discovery. We abstract the protocol, that the initiator, the intermediate nodes and the target are described as processes and all of these communication entities share the global clock. It achieves the effect of asymmetric key encryption through clock synchronization and time delay. Besides, we also propose an intruder model in which the intruder can eavesdrop, fake, intercept, learn and deduce the message from the protocol and the environment. Furthermore, by applying PAT [34], we verify the security properties of the Ariadne protocol model and we find that the fake routing attacks may be present in the protocol, which have been pointed out in [1,5]. Finally, we advocate that this suggested framework is also applicable to other protocols, based on the on-demand routing protocols, in which the route discovery process can be modeled as general processes. The main contributions of this paper are listed as follows:

- **Modeling.** A formal model for Ariadne Protocol is given using process algebra CSP. The communication entities of the protocol, including the initiator, the intermediate nodes and the target, are modeled as CSP processes, and we propose an intruder model and produce a CSP description of the specifications.
- **Analysis.** We analyze the whole process of the route discovery of the Ariadne protocol, adding a set of rules into the intruder model. We also give the analysis of the fake path existence in the case studies.
- **Verification.** The formal model is implemented in the model checking tool PAT. The security properties of Ariadne Protocol, i.e., Deadlock Freedom, Message Consistency, Node List Security, Fake Path Nonexistence and End-to-End Nodes Authentication, are verified by PAT. The verification results show that there is a defect in Ariadne Protocol, which may lead to fake routing attacks.

The rest of this paper is organized as follows. We introduce preliminaries about CSP and PAT in Section 2. An overview of the Ariadne secure routing

protocol is presented in Section 3. We formalize the protocol in Section 4 and in Section 5, based on the analysis of traces, we use model checker PAT to implement and verify the achieved model with five properties. In the Section 6, we discuss that the modeling approach presented in this paper is also applicable to other protocols, which are based on the on-demand routing protocols and have the route discovery process. We conclude the paper and present the future directions in Section 7.

2. Preliminaries

2.1. The CSP Method

Process algebra, as a representative of the formal methods, is to use algebraic approaches to study the communications of the concurrent systems. There are three typical calculus systems: Calculus of Communicating Systems (CCS) [24], Communicating Sequential Processes (CSP) and Algebra of Communicating Processes (ACP) [3]. In this subsection, we give a brief introduction to CSP (Communicating Sequential Processes) [8], which was proposed by C. A. R. Hoare in 1978. Nowadays, it has developed and already become one of the more mature process algebra formal method. It specializes in describing the interaction between concurrency systems using mathematical theories. Due to powerful expressive ability, CSP is widely applied in many fields. CSP processes are composed of primitive processes and actions.

The processes in this paper are defined using the following syntax. Here, P and Q represent processes which have alphabets $\alpha(P)$ and $\alpha(Q)$ to denote the set of actions that the processes can perform respectively. a and b stand for the atomic actions and c is the name of channel.

Skip	represents a process which does nothing but terminates successfully.
Stop	denotes that the process is in the state of deadlock and does nothing.
$P; Q$	the process performs P and Q sequentially.
$P \parallel Q$	describes the concurrent between P and Q.
$P[[a \leftarrow b]]$	indicates that a is replaced by b.
$a \rightarrow P$	the process first engages in action a, then the subsequent behavior is like P.
$c?x \rightarrow P$	the process gets a message through the channel c and assigns it to a variable x, then behaves like P.
$c!x \rightarrow P$	the process sends a message x using the channel c, then the behavior is like P.
$a \rightarrow P \Box b \rightarrow Q$	the process behaves like either P or Q and the selection is determined by the environment.
$P \backslash S$	stands for that the process behaves like P except all the actions in set S are concealed.

$P[\|X\|]Q$	the process represents that P and Q perform the concurrent events on the set X of channels.
$P \lhd b \rhd Q$	means if the condition b is true, the behavior is like P, otherwise, like Q.
$CHAOS(x)$	can perform any sequence of events from its alphabet x.

In the verification part, we also apply the trace model of CSP, which is composed of a set of traces, in describing a process. Here, the trace means the events that the process may perform. More details about CSP can be found in [4,8,30,31].

2.2. Process Analysis Toolkit

In this subsection, we give an overview of the verification tool PAT which will be applied in verifying our achieved model of the Ariadne protocol.

PAT (Process Analysis Toolkit) [14,15,16,35] is designed as an extensible and modularized framework for automatic system analysis based on CSP. It supports to specify and verify many different modeling languages and it has been used to model and verify a lot of different systems such as concurrent systems, real-time systems [37], probabilistic systems [38], web service models [36], sensor networks [44,45], and security protocols [20,40]. PAT can be applied in verifying varieties of properties such as deadlock-freeness, divergence-freeness, reachability, LTL properties with fairness assumptions, refinement checking and probabilistic model checking. We list some notations in PAT as follows:

1. $\#define\ N\ 0$ defines a global constant N which has the initial value 0.
2. $var\ msglist[N]$ defines an array named $msglist$ and the size of it is N.
3. $Channel\ c\ 5$ defines a communication channel and the capacity of it is 5.
4. $P = \{x = x + 1\} \rightarrow Skip$ defines an event that can be attached with assignment using which we can update the value of a global variable x.
5. $c!a.b \rightarrow P$ and $c?x.y \rightarrow P$ refer to sending message $a.b$ and receiving message from channel c respectively.

Besides, PAT can also describe the control flow structures, including $if - then - else$ and $while$, etc. More details about this tool can be found in [6,21,34].

3. Overview of the Ariadne Protocol

Ariadne, as an extension to the dynamic source routing (DSR) protocol, is composed of routing discovery and routing maintain. One of its main security goals is to prevent attackers or compromised nodes from tampering with uncompromised routes consisting of uncompromised nodes, and also prevent many types of Denial-of-Service attacks. And it also provides a property that no intermediate node can remove a previous node in the node list in the request or reply, which means that it can prevent a compromised node from removing a node

from the node list arbitrarily [10].

To achieve the security goal as mentioned, the Ariadne protocol applies three authentication mechanisms:

- TESLA protocol [26,27], which is used for route data authentication to certificate the integrity and authenticity of the routing message;
- End-to-End nodes authentication mechanism, which is used to verify the authenticity and freshness of the request and reply using the shared key;
- Per-hop hashing authentication mechanism, which is used to prevent an attacker from removing a node from the node list.

3.1. Notations and Assumptions

In this subsection, we give an overview of the notations and assumptions we will use in our paper. First, we introduce the following notations before describing the protocol:

- S stands for the initiator and D represents for the target;
- A and B are participants, such as the intermediate nodes;
- C denotes an internal node which is captured by the external intruders;
- * stands for all nodes in the whole network;
- K_{SD} and K_{DS}, used to certificate the identities of the communicating nodes, represent the secret MAC keys shared between S and D. The value of K_{SD} is equal to the value of K_{DS};
- $MAC_{K_{SD}}(M)$ denotes the MAC value calculated by the key K_{SD} and the message M.

Besides, during our formalization of the protocol, we also use some assumptions. We naturally inherit all of the assumptions of the Ariadne protocol and the TESLA protocol. In order to facilitate the modeling in the next section, we also list the following assumptions:

1. There is no efficient routing path between S and D in the local table of S.
2. Network is bidirectional [10], i.e., if A can receive the message from B, then B must be able to receive the message from A.
3. Each intermediate node has a TESLA one-way key chain and the keys in the chain are computed through the function $F(x)$. Each node firstly releases one key to all nodes in its broadcast range so that the follow-up nodes can use it to certificate the message. More details about TESLA protocol can be found in [26,27].
4. C, captured by intruders, can get initial information of all the nodes participating the process of route discovery. It can intercept or eavesdrop or fake messages passed between nodes and it can also be used as a normal node to join in the routing process.

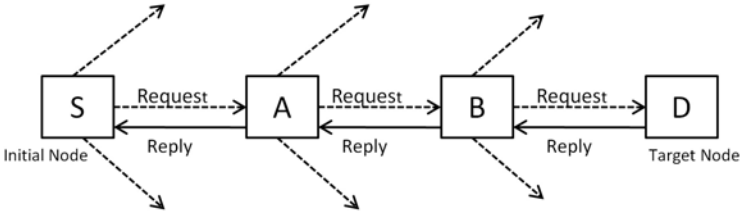


Fig. 2. An Example of the Route Discovery in Ariadne

Note that, security protocol attack models are divided into two types: active attacks and passive attacks. Most of the passive attacks refer to the attacks against the communication privacy, i.e., eavesdropping the data packets between the two communication entities, instead of the attacks against the routing protocols or network functions. On the other hand, active attacks mean that the intruders attack the routing protocols through tampering or faking routing messages to achieve the attack purpose. The sources of the active attacks are also divided into two types: the external intruders and internal intruders [17]. Although they may have the same intrusions, the security threats from the internal intruders are far greater than the ones posed by the external intruders, because the former can get more initial information of the protocol than the latter. Thus, in our paper, we focus more on the security threats brought by the internal intruders.

Using the notations and assumotions as mentioned, we next describe the process of Ariadne routing discovery and its authentication mechanism.

3.2. Description of the Ariadne Routing Discovery and Authentication Mechanism

When there is a packet needed to be sent to the destination, the initiator node first checks its local table. If there is already some path to the target, the initiator will send the packet along the existing path; Otherwise, it will start the route discovery. The process of the Ariadne protocol route discovery is divided into two parts: the initial node broadcasts a routing request to all of its neighbours and the target node sends a routing reply back to the initial node after it gets the request. In order to illustrate the process clearly, we give a simple topology example in Figure 2, where the dotted line means broadcast and the solid line stands for unicast.

Broadcasting the route request. According to Figure 2, S broadcasts a route request to all of its neighbors in the whole network. In Ariadne, a route request contains four fields such as $\langle msg_{req}, hash, node\,list, MAC\,list\rangle$. Here, msg_{req} stands for the route request message and $hash$ value is calculated by a per-hop hash function $H(x)$ except the first element which is computed as $MAC_{k_{SD}}(msg_{req})$.

Through the per-hop hash function $H(x)$, Ariadne protocol ensures that a malicious node cannot remove the intermediate node or modify the order of the node list arbitrarily. $node\ list$ and $MAC\ list$ store the addresses and the MAC values respectively, which are empty initially. msg_{req} also has five fields, $\langle request, initiator, target, id, time\ interval\rangle$. Here we ignore the details of the msg_{req} for simplicity. The process of the broadcasting is shown in Table 1.

Table 1. Broadcasting the Route Request

$S:$	$msg_{req} = (request, S, D, id_{SD}, t_i)$
	$h_S = MAC_{K_{SD}}(msg_{req})$
$S \rightarrow *:$	$\langle msg_{req}, h_S, (), ()\rangle$
$A:$	$h_A = H[A, h_S]$
	$M_A = MAC_{K_{A_{t_i}}}(msg_{req}, h_A, (A), ())$
$A \rightarrow *:$	$\langle msg_{req}, h_A, (A), (M_A)\rangle$
$B:$	$h_B = H[B, h_A]$
	$M_B = MAC_{K_{B_{t_i}}}(msg_{req}, h_B, (A, B), (M_A))$
$B \rightarrow *:$	$\langle msg_{req}, h_B, (A, B), (M_A, M_B)\rangle$

When the intermediate nodes receive the route request, they will first verify the authenticity of the route request message and the effectiveness of the key that the previous node uses. If the TESLA key has already been released, the node will discard this message and send an error message back to the initial node. Otherwise, the current node adds its address, hash value and the MAC value computed by its own TESLA key into the request message and rebroadcasts it.

Unicasting the route reply. After D receives the route request message, it checks the consistency of the $node\ list$ and also tests whether any key the nodes use has already been released within the specified time or not. Only in the case that all the conditions we have mentioned above are satisfied will D accept the route request and send a route reply back to S along the reverse order of the nodes in the $node\ list$.

In Ariadne, a route reply contains three fields, $\langle msg_{rep}, target\ MAC, key\ list\rangle$. Here, $target\ MAC$ stands for the MAC value of the target node and $key\ list$ is empty initially. Besides, msg_{rep} consists of six fields, $\langle reply, target, initiator, time\ interval, node\ list, MAC\ list\rangle$. Table 2 illustrates the process of unicasting the route reply.

Table 2. Unicasting the Route Reply

$D:$	$msg_{rep} = (reply, D, S, t_i, (A, B), (M_A, M_B))$
	$M_D = MAC_{k_{DS}}(msg_{rep})$
$D \rightarrow B:$	$\langle msg_{rep}, M_D, ()\rangle$
$B \rightarrow A:$	$\langle msg_{rep}, M_D, (K_{B_{t_i}})\rangle$
$A \rightarrow S:$	$\langle msg_{rep}, M_D, (K_{B_{t_i}}, K_{A_{t_i}})\rangle$

When an intermediate node receives the route reply, it caches the message until its TESLA key is released. Then the node adds its own key into the $key\ list$ and sends the message to the last node. When S receives the reply, it will check the correctness of each key and each MAC value respectively. Besides, it will also verify the MAC value of D. Only when there is no error in the process of the validation would S accept the route reply and cache the path in its local table. Otherwise, S will discard the reply.

4. Formalizing the Ariadne Protocol

In this section, we use CSP to model the route discovery of the Ariadne protocol. Firstly, we define the sets and channels that we would use below.

- We assume the existence of the set **Initiator** of initiators, the set **Target** of targets.
- The set **Node** stands for the intermediate nodes and **Intruder** represents the internal nodes captured by the malicious nodes.
- The set **SharedKey** contains the keys shared between the initiator and the target.
- The set **Key** involves the TESLA keys that the intermediate nodes use.

In addition, there is another set **MSG**, which stores all the messages passing in the whole route discovery process. We also define four types of the messages as follows:

$$MSG1 =_{df} \{msg_{req}.h_s.S.*|h_s = MAC(K_{SD}, msg_{req}), K_{SD} \in SharedKey, S \in Initiator, * \in Node \cup Target\}$$

$$MSG2 =_{df} \{msg_{req}.h_{X_i}.(..X_j..X_i).(..M_{X_j}..M_{X_i}).X_i.*|* \in Node \cup Target, M_{X_i} = MAC(K_{X_{i_{t_i}}}, msg_{req}, h_{X_i}, (..X_j..X_i), (..M_{X_j}..M_{X_{i-1}})), h_{X_i} = H(X_i, h_{X_{i-1}}), K_{X_{i_{t_i}}} \in Key, X_j, X_i \in Node, i \in N\}$$

$$MSG3 =_{df} \{msg_{rep}.M_D.D.X_i|M_D = MAC(K_{DS}, msg_{rep}), K_{DS} \in SharedKey, i \in N, X_i \in Node, D \in Target\}$$

$$MSG4 =_{df} \{msg_{rep}.M_D.(..K_{X_{j_{t_i}}}..K_{X_{i_{t_i}}}).X_i.S|M_D = MAC(K_{DS}, msg_{rep}), K_{X_{j_{t_i}}}, K_{X_{i_{t_i}}} \in Key, i \in N, X_j, X_i \in Node, S \in Initiator\}$$

$$MSG =_{df} MSG1 \cup MSG2 \cup MSG3 \cup MSG4$$

Here, the specific meaning of each message set can be explained as follows:

1. $MSG1$ represents the set of broadcast messages which are sent by the initiator to the intermediate nodes or the target. Here, h_s stands for the MAC value, which is computed by the key K_{SD}, shared by the initiator and the target, and the request message msg_{req}.

2. The set of $MSG2$ stores the messages that are broadcasted by the intermediate nodes. Any intermediate node receives the request message, it will compute its own hash value and MAC value, then the node will modify the request message using these two values and rebroadcast the request message to the next node.
3. The messages in the set of $MSG3$ stand for the reply messages which are sent by the target to some intermediate node. After the target node receives the request message, it will check the correctness of each value in the message. If all the values are valid, the target node will unicast the reply message according to the reverse order of the intermediate nodes in the node list.
4. The set of $MSG4$ holds the reply messages that are sent back to the initial node.

We use four channels to model the communication in the process of the Ariadne protocol: $Broadcast$, $ComX_iX_j$, $Intercept$, $Fake$.

- $Broadcast$: it is used to broadcast and re-broadcast the message.
- $ComX_iX_j$: it denotes the standard communication between nodes X_i and X_j. Here, nodes include the initiator, the intermediate nodes and the target.
- $Intercept$: the intruder uses it to intercept the information from normal communications.
- $Fake$: it is used by the intruder to send messages which are modified to the normal nodes.

In addition, we also define another two channels: $Session$ and $Fake\_session$, which represent a successful communication and a successful intrusion respectively. The declaration of the channels are as follows:

Channel $Broadcast$, $ComX_jX_i$, $Intercept$, $Fake$: MSG
Channel $Session$, $Fake\_session$: Initiator.Target

Figure 3 illustrates the communications between nodes using channels.

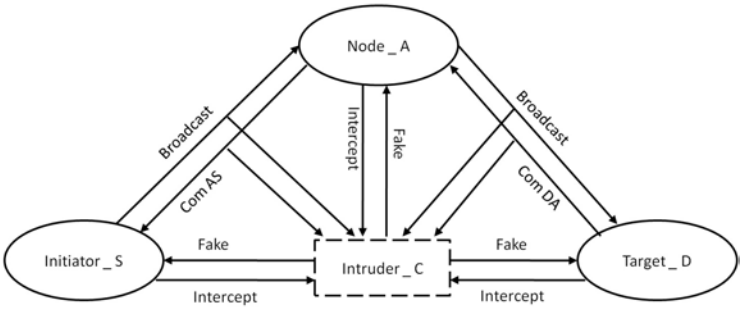


Fig. 3. Communications between Nodes Using Channels

4.1. Clock

The clock is the shared communication entity among the initiator, intermediate nodes and the target. Through clock synchronization and the delaying of releasing the key, the protocol which uses the TESLA broadcasting authentication mechanism achieves the desired effect of asymmetric key encryption.

$Clock(i) =_{df} (tick \rightarrow Clock(i+1))\Box(time?request \rightarrow time!i \rightarrow Clock(i))$
where $i >= 0 \wedge i \in \mathbb{N}$.

Note that, if $Clock$ receives a request from the channel $time$, it will return the current time i.

4.2. Initiator

In the whole process of route discovery, initiator will broadcast the route request and receive the route reply. Besides, it also has to check the validity of each key in the key list and the correctness of the MAC value of the target. Without considering the intruders, we model the behaviors of the initiator node as follows:

$Initiator_1(S, T_0, t_{int}) =_{df} time!request \rightarrow time?T_{S_0} \rightarrow Stop \triangleleft$
$(T_{S_0} > (T_0 + (T+1) * t_{int}) | T_{S_0} < T_0) \triangleright Initiator_2(S, D, K_{SD}, K_{DS})$

The time beginning to send route request is T_{S_0} which must be within the time period from T_0 to T_{T+1}. The time period is divided into $T+1$ small time intervals with a length of t_{int} for each one.

$Initiator_2(S, D, K_{SD}, K_{DS}) =_{df}$
$|||_{X_i \in Node \wedge i \in [1,n] \wedge n \in N}(Broadcast!msg_{req}.MAC(K_{SD}, msg_{req}).S.X_i \rightarrow$
$ComX_iS?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{t_i}}.X_i.S \rightarrow Stop \triangleleft$
$(MAC(K_{SD}, msg_{rep})! = MAC(K_{DS}, msg_{rep})) \triangleright CheckKeyValid_1(\overrightarrow{NL}, \overrightarrow{K_{t_i}})$
$\rightarrow Session.S.D) \rightarrow Initiator_1(S, T_0, t_{int})$

$\overrightarrow{K_{t_i}}$ and $\overrightarrow{NL}$ stand for the key list and the node list respectively. When the initiator receives the reply message, it will check the correctness of the MAC value of the target firstly, through End-to-End nodes authentication mechanism. Then, the process of $CheckKeyValid_1$ is used to check the validation of any key in the key list by the initiator and we give its definition as follows:

$CheckKeyValid_1(\overrightarrow{NL}, \overrightarrow{K_{t_i}}) =_{df} Skip \triangleleft (len(\overrightarrow{K_{t_i}}) == 0) \triangleright$
$(CheckKeyValid_1(tail(\overrightarrow{NL}), tail(reverse(\overrightarrow{K_{t_i}}))) \triangleleft$
$(keymap(head(\overrightarrow{NL}) == F^{i-j}(head(reverse(\overrightarrow{K_{t_i}}))) \triangleright Stop)))$

Here, len shows the length of the list. $head$ and $tail$ return the first element and the remainder of a list respectively. $keymap$ is a function to get the authenticated key at time t_j according to the node id and through $F^{t_i - t_j}(K_{t_i})$ one can get the key K_{t_j} at time t_j. Then, we apply this equation to the received key

value to determine if the computed value matches a previous known authentic key value at time t_j on the key chain. Here, $reverse$ is also a function to reverse the elements in the key list $\overrightarrow{K_{t_i}}$.

Indeed, we must allow the possibility of intruder actions, such as intercepting the request messages and faking the reply messages. We model this situation via the renaming of CSP:

$$
\begin{aligned}
&Initiator(S, T_0, t_{int}) =_{df} Initiator_1(S, T_0, t_{int}) \\
&\quad [[Broadcast!msg_{req}.MAC(K_{SD}, msg_{req}).S.X_i \\
&\qquad \leftarrow Broadcast!msg_{req}.MAC(K_{SD}, msg_{req}).S.X_i, \\
&\quad Broadcast!msg_{req}.MAC(K_{SD}, msg_{req}).S.X_i \\
&\qquad \leftarrow Intercept!msg_{req}.MAC(K_{SD}, msg_{req}).S.X_i, \\
&\quad ComX_iS?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{t_i}}.X_i.S \\
&\qquad \leftarrow ComX_iS?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{t_i}}.X_i.S, \\
&\quad ComX_iS?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{t_i}}.X_i.S \\
&\qquad \leftarrow Fake?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{t_i}}.X_i.S, \\
&\quad Session.S.D \leftarrow Session.S.D, \\
&\quad Session.S.D \leftarrow Fake\_session.S.D]]
\end{aligned}
$$

4.3. Node

The intermediate nodes, in the whole process of the route discovery, have four types of actions: receive the broadcasting request messages from the initiator or the last intermediate node, rebroadcast the modified request messages to the next node, get the unicasting reply messages and pass the reply messages to the intermediate node or the initiator. Before modeling the actions for the intermediate nodes, the definition of the process $Wait(t)$ will be listed below:

$$Wait(t) =_{df} Skip \triangleleft (t == 0) \triangleright (tick \rightarrow Wait(t-1)) \qquad \text{where } t >= 0$$

The process of $Wait(t)$ is used to represent waiting for t time. If the intermediate node receives the reply message but its own TESLA key has not released, it should cache the reply message and wait for a few time until its own key is released. $Wait(t)$ process synchronizes with the global timer and countdowns the time t until it equals to 0. We will give the CSP model ignoring the intruder firstly.

Because the intermediate node uses the TESLA broadcasting authentication protocol for route data authentication, after it receives the request message, it checks whether the key the last node used has already been released or not. If the key has already been released, the node will discard the request, otherwise, it will compute its own hash value and MAC value, and rebroadcast the modified request message to the next nodes. In the model below, X_l stands for the last node, X_i for the current node and X_n for the next node.

$$
\begin{aligned}
&Node_1(S, X_l, X_i, K_{X_{i_{t_i}}}, \delta, T_{S_0}, t_{int}) =_{df} (|||_{X_n \in Node \wedge n \in N \wedge X_n != X_i}(\\
&\quad Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.X_i \rightarrow time!request \rightarrow \\
&\quad time?T_{S_1} \rightarrow CheckkeyValid_2(T_{S_0}, T_{S_1}, S, X_l, X_i, \delta, t_{int}) \rightarrow \\
&\quad Broadcast!msg_{req}.\overrightarrow{h_{X_i}}.\overrightarrow{NL_{X_i}}.\overrightarrow{MAC_{X_i}}.X_i.X_n \rightarrow \\
&\quad ComX_nX_i?msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{X_{n_{t_i}}}}.X_n.X_i \rightarrow \\
&\quad time!request \rightarrow time?T_{S_2} \rightarrow WaitKeyReleased(T_{S_0}, T_{S_2}, S, X_i, \delta, t_{int}) \rightarrow \\
&\quad ComX_iX_l!msg_{rep}.MAC(K_{DS}, msg_{rep}).\overrightarrow{K_{X_{i_{t_i}}}}.X_i.X_l) \rightarrow \\
&\quad Node_1(X_l, X_i, K_{X_{i_{t_i}}}, \delta, T_{S_0}, t_{int}))
\end{aligned}
$$

$\overrightarrow{h_{X_i}}$ stands for the hash chain constructed using the per-hop hash function $H(x)$, and $\overrightarrow{MAC_{X_i}}$ stands for the current MAC list. $CheckKeyValid_2$ is a process to check the TESLA key the last node used is released or not, and $WaitKeyReleased$ is another process used for node X_i itself to wait until its TESLA key is released at some time. Here, δ represents the number of the time intervals from starting using the key to releasing it. The definitions of the processes $CheckKeyValid_2$ and $WaitKeyReleased$ are given below:

$$
\begin{aligned}
&CheckKeyValid_2(T_{S_0}, T_{S_1}, S, X_l, X_i, \delta, t_{int}) =_{df} \\
&\quad Stop \lhd \\
&\quad ((T_{S_1} + GetTimeDif(X_l, X_i)) >= (T_{S_0} + GetTimeDif(S, X_l) + \delta * t_{int})) \\
&\quad \rhd Skip
\end{aligned}
$$

Here, $GetTimeDif$ is a function to get the time differences between two nodes. According to the TESLA broadcasting authentication protocol, the key the node uses will be released after $\delta * t_{int}$ time from the time it begins to be used.

$$
\begin{aligned}
&WaitKeyReleased(T_{S_0}, T_{S_2}, S, X_i, \delta, t_{int}) =_{df} \\
&\quad Wait(T_{S_0} + GetTimeDif(S, X_i) + \delta * t_{int} - T_{S_2}) \lhd \\
&\quad (T_{S_2} < (T_{S_0} + GetTimeDif(S, X_i) + \delta * t_{int})) \rhd Skip
\end{aligned}
$$

The intermediate node uses the process above to decide whether its key released or not, and it should cache the reply data packet until its own TESLA key is released. Here, we can also use renaming to model the behaviors of the node so as to consider the actions of the intruder as process of $Node$. The intruder may fake or intercept the request and the reply messages, thus we list the detailed model as follows:

$$
\begin{aligned}
&Node(S, X_l, X_i, K_{X_{i_{t_i}}}, \delta, T_{S_0}, t_{int}) =_{df} Node_1(S, X_l, X_i, K_{X_{i_{t_i}}}, \delta, T_{S_0}, t_{int}) \\
&\quad [[Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.X_i \\
&\qquad \leftarrow Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.X_i, \\
&\quad Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.X_i \\
&\qquad \leftarrow Fake?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.X_i, \\
&\quad Broadcast!msg_{req}.\overrightarrow{h_{X_i}}.\overrightarrow{NL_{X_i}}.\overrightarrow{MAC_{X_i}}.X_i.X_n \\
&\qquad \leftarrow Broadcast!msg_{req}.\overrightarrow{h_{X_i}}.\overrightarrow{NL_{X_i}}.\overrightarrow{MAC_{X_i}}.X_i.X_n,
\end{aligned}
$$

$$
\begin{aligned}
&Broadcast!msg_{req}.\overrightarrow{h_{X_i}}.\overrightarrow{NL_{X_i}}.\overrightarrow{MAC_{X_i}}.X_i.X_n\\
&\quad \leftarrow Intercept!msg_{req}.\overrightarrow{h_{X_i}}.\overrightarrow{NL_{X_i}}.\overrightarrow{MAC_{X_i}}.X_i.X_n,\\
&ComX_nX_i?msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{n_{t_i}}}}.X_n.X_i\\
&\quad \leftarrow ComX_nX_i?msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{n_{t_i}}}}.X_n.X_i,\\
&ComX_nX_i?msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{n_{t_i}}}}.X_n.X_i\\
&\quad \leftarrow Fake?msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{n_{t_i}}}}.X_n.X_i,\\
&ComX_iX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{i_{t_i}}}}.X_i.X_l)\\
&\quad \leftarrow ComX_iX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{i_{t_i}}}}.X_i.X_l),\\
&ComX_iX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{i_{t_i}}}}.X_i.X_l)\\
&\quad \leftarrow Intercept!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{X_{i_{t_i}}}}.X_i.X_l)]]
\end{aligned}
$$

4.4. Target

After the target receives the route request, it checks whether there is any key the nodes use has been released or not through the process $CheckKeyValid_3$ and it also checks the MAC value of the route request using K_{DS}. If there is no error, it will unicast a route reply according to the reverse order of the nodes in the node list. Here, the key list $\overrightarrow{K_{t_i}}$ is an empty list and $\overrightarrow{\Delta t}$ is a list which holds the time difference between every two nodes.

$$
\begin{aligned}
&Target_0(X_l,D,K_{DS},\delta,t_{int},T_{S_0},\overrightarrow{\Delta t}) =_{df}\\
&\quad Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.D \rightarrow\\
&\quad time!request \rightarrow time?T_{S_4} \rightarrow\\
&\quad CheckKeyValid_3(\overrightarrow{NL_{X_l}},\delta,t_{int},T_{S_0},\overrightarrow{\Delta t}) \rightarrow\\
&\quad ComDX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{t_i}}.D.X_l \rightarrow\\
&\quad Session.S.D \rightarrow Target_0(X_l,D,K_{DS},\delta,t_{int},T_{S_0},\overrightarrow{\Delta t})
\end{aligned}
$$

In addition, with respect to the intruder, it can intercept the reply message or fake the request message. We model the target via renaming as follows:

$$
\begin{aligned}
&Target(X_l,D,K_{DS},\delta,t_{int},T_{S_0},\overrightarrow{\Delta t}) =_{df}\\
&\quad Target_0(X_l,D,K_{DS},\delta,t_{int},T_{S_0},\overrightarrow{\Delta t})\\
&\quad [[Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.D\\
&\quad \leftarrow Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.D,\\
&\quad Broadcast?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.D\\
&\quad \leftarrow Fake?msg_{req}.\overrightarrow{h_{X_l}}.\overrightarrow{NL_{X_l}}.\overrightarrow{MAC_{X_l}}.X_l.D,\\
&\quad ComDX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{t_i}}.D.X_l\\
&\quad \leftarrow ComDX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{t_i}}.D.X_l,\\
&\quad ComDX_l!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{t_i}}.D.X_l\\
&\quad \leftarrow Intercept!msg_{rep}.MAC(K_{DS},msg_{rep}).\overrightarrow{K_{t_i}}.D.X_l,\\
&\quad Session.S.D \leftarrow Session.S.D,
\end{aligned}
$$

$Session.S.D \leftarrow Fake\_session.S.D]]$

4.5. Intruder

In this paper, we also propose an intruder model. Here, the intruder is regarded as a process which can intercept, eavesdrop and fake the message. Through learning, the intruder can get much information about the route discovery. We define a set $Fact$ explaining all the facts it learns.

$$\begin{aligned} Fact =_{df}\ & Initiator \cup Node \cup Target \cup Key \cup SharedKey \cup MSG \\ & \cup\{F(K)|K \in Key\} \\ & \cup\{MAC(K,msg)|K \in SharedKey, msg \in MSG\} \\ & \cup\{H(node,H^{'})|node \in Node\} \\ & \cup\{MAC_i(K,msg,H_i,(..node_j..node_i),(..MAC_j..MAC_{i-1}))| \\ & \quad K \in Key, msg \in MSG, node_j, node_i \in Node\} \end{aligned}$$

The intruder can learn facts from all the sets above. Besides, the intruder node can also deduce some facts from what it has known. We denote $I \mapsto f$ to represent that the intruder deduces new fact f from the known set I.

1. $\{F(K_{X_i})\} \mapsto K_{X_{i-1}}, K_{X_{i-2}}, .., K_{X_0}$
2. $\{MAC(K,msg)\} \mapsto MAC(K,msg)$
3. $\{MAC_i(K,msg,H_i,(..node_j..node_i),(..MAC_j..MAC_{i-1}))\} \mapsto$
   $MAC_i(K,msg,H_i,(..node_j..node_i),(..MAC_j..MAC_{i-1}))$
4. $\{H(node,H^{'})\} \mapsto H(node,H^{'})$
5. $I \mapsto f \wedge I \subseteq \mathbf{I} \Rightarrow \mathbf{I} \mapsto f$

Here, the first deducing rule means that in the TESLA one-way key chain authentication protocol, the intruder can deduce all the previous key values through the one-way key function and any one key. The following three rules stand for the intruder can reason the MAC values and the hash values from the known sets. The last one represents that if the intruder can deduce one new fact f from the known set I, it also can deduce the fact f from the set $\mathbf{I}$, which is bigger than I. As stated in [10], we also assume that the intruder knows the identity of each node so that it can get some information from the fact without deducing. Through the function $info$, the intruder can get various parts of the message. For example:

$$\begin{aligned} & info(msg_{req}.h_S.S.*) = \{msg_{req}, h_S, S, *\} \\ & info(msg_{req}.h_{X_i}.(..X_j..X_i).(..MAC_{X_j}..MAC_{X_i}).X_i.*) = \\ & \quad \{msg_{req}, h_{X_i}, .., X_j, .., X_i, ..MAC_{X_j}, .., MAC_{X_i}, X_i, *\} \\ & info(msg_{rep}.MAC_D.D.X_i) = \{msg_{rep}, MAC_D, D, X_i\} \end{aligned}$$

$$info(msg_{rep}.MAC_D.(..K_{X_{j_{t_i}}}..K_{X_{i_{t_i}}}).X_i.S) =$$
$$\{msg_{rep}, MAC_D, .., K_{X_{j_{t_i}}}, .., K_{X_{i_{t_i}}}, X_i, S\}$$
where $msg_{req}, msg_{rep} \in MSG,\ S \in Initiator,\ * \in Node \cup Target,$
$X_j, X_i \in Node,\ D \in Target,\ K_{X_{j_{t_i}}}, K_{X_{i_{t_i}}} \in Key$

In order to model the behaviors of the intruder, we define another channel $deduce$, through which the intruder can deduce some new facts from the set of the facts it knew. The declaration of channel $deduce$ is:

Channel $deduce : Fact.P\{Fact\}$.

Here, P stands for the power set of the $Fact$. The model of the intruder is as follows:

$Intruder_0(I) =_{df}$
$\Box_{m \in MSG} Broadcast.m \rightarrow Intruder_0(I \cup info(m))$
$\Box$
$\Box_{m \in MSG} ComX_iX_j.m \rightarrow Intruder_0(I \cup info(m))$
$\Box$
$\Box_{m \in MSG} Intercept.m \rightarrow Intruder_0(I \cup info(m))$
$\Box$
$\Box_{m \in MSG} Fake.m \rightarrow Intruder_0(I)$
$\Box$
$\Box_{f \in Fact, f \notin I, I \mapsto f} deduce.f.I \rightarrow Intruder_0(I \cup \{f\})$

We hide the $deduce$ channel because of the internal actions, then we get the model of the intruder:

$$Intruder(I) =_{df} Intruder_0(I) \setminus [|\{deduce\}|]$$

4.6. System and Specification

The whole system can be modeled as the parallel composition of the initiator, the intermediate nodes and the target. All these communication entities share the same clock. First we consider the system without intruder.

$INITIATOR =_{df} Clock(0)[|\{time\}|]Initiator(S, T_0, t_{int})$
$NODE =_{df} Clock(0)[|\{time\}|]Node(S, X_l, X_i, K_{X_{i_{t_i}}}, \delta, T_{S_0}, t_{int})$
$TARGET =_{df} Clock(0)[|\{time\}|]Target(X_l, D, K_{DS}, \delta, t_{int}, T_{S_0}, \overrightarrow{\Delta t})$
$SYSTEM_0 =_{df} INITIATOR[|\{Broadcast, ComX_iS, Session\}|]NODE$
$[|\{Broadcast, ComDX_i, Session\}|]TARGET$

Then, we add the intruder into the whole system.

$$SYSTEM =_{df} SYSTEM_0[|INTRUDER\_ALPH|]INTRUDER(S, D, C, K_{C_{t_i}})$$

$$INTRUDER\_ALPH =_{df} \{|Broadcast, ComX_iX_j, Intercept, Fake, Session, Fake\_session|\}.$$

Ariadne aims to preventing a malicious node from compromising the route. The initiator or the target cannot accept any message modified by the intruder. Here, we define the specification for the security property of the Ariadne protocol as:

$$SPEC =_{df} CHAOS(\sum -\{|Fake\_session.S.D|\}).$$

Note that $\sum$ stands for the set of all the events. As we mentioned in Section 4, the channel $Fake\_session$ represents a successful intrusion between the initiator and the target. In Section 2, $CHAOS(x)$ is explained that the process can perform any sequence of events from its alphabet x. Thus, the whole specification means that the process can perform all the events except the ones on the set of channel $Fake\_session$. Some specific security properties will be listed in the next section.

5. Verification in PAT

In this section, we use PAT to implement our CSP model of the Ariadne protocol. We have already given a brief introduction to PAT in Section 2.

5.1. The Ariadne Protocol in PAT

Here, we implement three cases of our model using PAT. Case I is a basic instance, ignoring the intruder, with a simple topology to show the basic process of route discovery of the Ariadne protocol. Case II and Case III are more complex with two different types of intruders: Case II with the Active-1-1 intruder and Case III with the Active-1-2 intruder. Fake routing attacks may be present in these two cases. Before we implement these three cases in PAT, we first define four other functions in a new $C\sharp$ library as follows:

- $MACValue$ is used only for the initiator and the target to compute their MAC values using the shared key and the message.
- $HashValue$ is used to compute the hash values for the intermediate nodes through function $H(x)$.
- $mediaMACValue$ is used for the intermediate nodes to compute the MAC values using their corresponding TESLA keys.
- $KeyValue$ is a function to compute the node's TESLA key at some time on the TESLA chain.

We also need some significant channels and variables, e.g. $broadcast$ stands for the broadcasting channel, N represents the number of the intermediate nodes in our case, $msgreq$ stands for the request message, $idlist[N+2]$ is a list

that stores the identity for each node, $msgreply[N+1][2]$ is a two-dimensional array that stores the reply messages, $distancelist$ is also a list recording the distance between each two nodes. We give the declarations of them as follows:

```
#define N 4;
channel  broadcast (N+1)*N;
#define msgreq 10;
var msglist[N+1][4];
var idlist[N+2] = [0,1,2,3];
var msgreply[N+1][2];
var distancelist[N+2][N+2]=
                [0,1,2,3,-1,0,1,2,-1,-1,0,1,-1,-1,-1,0];
```

Case I: We implement the basic process of route discovery without intruder. We assume that there are only four nodes such as S, A, B and D which have the same topology as Figure 2, which we have already mentioned in Section 3.

Here, we give the relevant codes in PAT to show how the initiator node sends and receives messages. The initiator S broadcasts the request message to all of its neighbors and it also receives the reply message and does some appropriate checks such as checking the time validation which is interpreted by the process $CheckInitTime(clock)$. We give the PAT code of the processes $SendInits$, $SendInit$ and $SendInit2$ as follows:

```
SendInits(sender, content) = ||| receiver:{1..N+1} @
         (SendInit(sender, content, receiver));
SendInit(sender, content, receiver) =
          ifa ((sender == receiver)||
          (idlist[sender]>idlist[receiver])||
          distancelist[sender][receiver]!=1){ Skip }
          else
          { time!true -> time?x -> {clock = x} ->
            CheckInitTime(clock);
            SendInit2(sender, content, receiver)
          };
SendInit2(sender, content, receiver) =
          {MACInit = call(MACValue,KeySD,msgreq)}->
           broadcast!sender.receiver.content.MACInit->
           {msglist[sender][0] = sender;
            msglist[sender][1] = content;
            msglist[sender][2]=MACInit;
            msglist[sender][3]=0
           } ->Skip;
```

In the above codes, some parameters from the model are defined as the global variables which we have already mentioned before. Here, the process of the whole system is represented:

```
System() = Clock(0) |||( Initiator(1)|||Nodes() ||| Target(3));
```

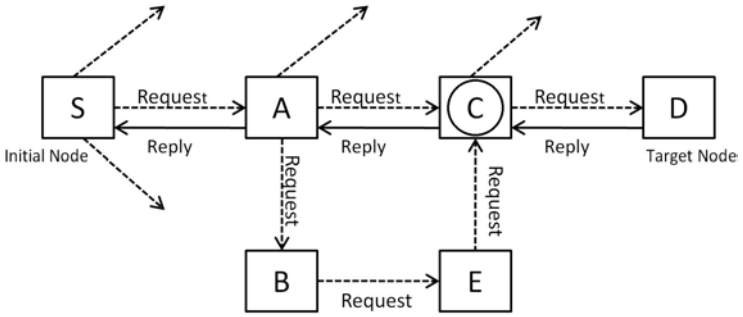


Fig. 4. Case II with Active-1-1 Intruder

Case II: Considering the Active-1-1 intruder, the intruder owns only one compromised node. We add the process of the intruder into the system and the topology is shown in Figure 4, in which we use a circle to identify the intruder node. In the code, the id corresponding to the node (S, A, B, E, C, D) is $(0, 1, 2, 3, 4, 5)$.

When the intruder C receives the message from A, it will wait until the message from E arrives. Then C will change the message especially the node list. After C gets the reply from D, it will cache the reply message until its own TESLA key and the TESLA key of node B have been released, then it will send the reply to A directly. In fact, there is a fake path that S does not recognize. This attack has been mentioned in [5], here we implement it using PAT and we will discuss the details in the subsection Result Analysis. We give the relevant codes about the actions of the intruder in this case, shown below:

```
Intruder(i) = broadcast?sender.i.msg.mac ->
              save.i{intruderlist[j][0]=sender;
              intruderlist[j][1]=i;intruderlist[j][2]=msg;
              intruderlist[j][3]=mac;j=j+1;
              cnt_intruder = cnt_intruder + 1}-> Change(i,j);
              ifa (cnt_intruder >= 2) {Skip}
              else {Intruder(i)};
Change(i,k) = ifa(k==2)
              {change.i{msglist[i-1][0]=i;
               msglist[i-1][2]=
                      call(HashValue,i,msglist[i-2][2]);
               msglist[i-1][3]=
                      call(mediaMACValue,KeyTi[i-1],
                      msglist[i-1][1],msglist[i-1][2],i,
                      msglist[i-2][3])}->
               Sendmedias(i,msglist[i-1][1],msglist[i-1][2]);
               ReceiveReply(i)}
              else {Skip};
```

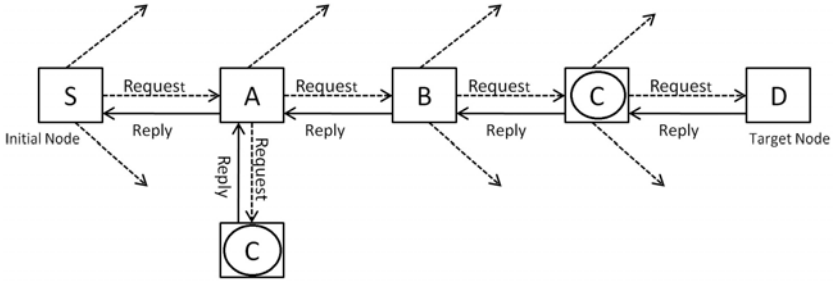


Fig. 5. Case III with Active-1-2 Intruder

Case III: Here, we consider the Active-1-2 intruder, which means that the intruder controls two corrupted nodes but they use the same compromised identifier. The topology is shown in Figure 5 and we use a circle to identify the intruder nodes. The id corresponding to the node (S, A, B, C, D) is $(0, 1, 2, 3, 4)$. When the intruder C receives the request from A, it saves the message into his private list, which the other corrupted node can also access. Then, the intruder whose identifier is also C will receive the message from B, it will change the node list, sending it to D. After C receives the reply message from D, it will also put it into its own list, then the other node whose identity is also named C will send the message back to A ignoring the intermediate node B. Relevant codes are represented below:

```
Intruder(i) = broadcast?sender.i.msg.mac ->
              save.i{intruderlist[j+1][0]=sender;
              intruderlist[j+1][1]=i;intruderlist[j+1][2]=msg;
              intruderlist[j+1][3]=mac} ->Change(i);
Change(i) = change.i{msglist[i-1][0]=i;
            msglist[i-1][2]=
                    call(HashValue,i,msglist[i-1][1]);
            msglist[i-1][3]=
                    call(mediaMACValue,KeyTi[i-1],
                    msglist[i-1][1],msglist[i-1][2],
                    i,intruderlist[0][3])}->
            Sendmedias(i,msglist[i-1][1],msglist[i-1][2]);
            ReceiveReply(i);
```

5.2. Verification

Based on the trace analysis, in our paper, we mainly list five properties as follows:

Property 1: Deadlock Freedom

Deadlock Freedom means that there is no state with no further move except

waiting for some sources occupied by other states. The property of Deadlock freedom can be formalized as follows:

$$\forall i \in (Initiator \cup Node \cup Target) \bullet$$
$$(clock <= TimeMax \wedge i.send == true) \Longrightarrow$$
$$(i.receive == true \wedge num(send) == num(receive))$$

Within the specific time, if there is some node sending message through the channel, there must be some node waiting to receive the message and the number of the sent message is equal to the number of the received message.

Property 2: Message Consistency

Message Consistency stands for no changes happened on the messages in the whole process of the protocol. Here, this property is explained below:

$$\exists m, n \in \mathbb{N} \bullet$$
$$(((request == msglist[m][1]) \Longrightarrow (\forall j\ request == msglist[j][1])) \wedge$$
$$((reply == replylist[n][0]) \Longrightarrow (\forall k\ reply == reply[k][0])))$$

All the request messages, especially the message received by the target node, must be consistent with the request sent by the initiator node. And the reply message received by all the nodes, including the initiator node, is the same with the message sent by the target node.

Property 3: Node List Security

Node List Security represents the security of the node list, that is the node list cannot be changed arbitrarily. We use the first order logic language to describe this property:

$$\forall i, j \in NID \bullet$$
$$(nodelist[i] == requestlist[i][0]) \wedge (replylist[j+1][2] == nodelist[j])$$
where $NID =_{df} 0..N$

The Ariadne protocol aims to prevent a malicious node from compromising the route. It wants to ensure the security of the node list that no intruder can change the order of the nodes, or add, remove any node from the node list. So the initiator receives the node list, which must be the same with that the target node receives.

Property 4: Fake Path Nonexistence

Fake Path Nonexistence is one of the most important properties we discussed in our paper. It means that there exists no fake path in the whole process of the route protocol. This property can be formalized as follows:

$$\forall k \in NID \bullet (nodelist[k] == msglist[k][0]) \wedge$$
$$(distance[msglist[k][0]][msglist[k+1][0] == 1)$$

where $NID =_{df} 0..(N-1)$

If the initiator receives the node list, it will check the distance of each node in the node list. The distance between neighbor nodes being not equal to one means that there may exist a fake path in the node list.

Property 5: End-to-End Nodes Authentication

End-to-End Nodes Authentication means that the end nodes also have some functions to certificate the correctness of the key value and the MAC value. Here, we mainly consider about the authentication of the initiator. This property is explained below:

$$\forall k \in NID \bullet (KeyValue(msgreply[k][1]) == Key_{t_j}[k]) \wedge (MACValue(Key_{SD}, msgreply[k][0]) == MACTar)$$

where $NID =_{df} 0..(N-1)$

In Ariadne protocol, when the initiator receives the reply form the last node, it will check the MAC value of the target and each key in the key list according to the TESLA protocol.

The assertions of these properties can be found in Table 3. According to the order of the properties as mentioned above, the following assertions describe Deadlock Freedom, Message Consistency, Node List Security, Fake Path Nonexistence and End-to-End Nodes Authentication respectively.

Table 3. Assertions of Properties

```
#define goal1 (!((clock ≤ TimeMax)&&(countsend !=0))||
               ((countreceive !=0)&&(countsend==countreceive)));
#assert System() reaches goal1;

#define goal2 ((msgreq==msgReced)&& (msgrep==msgRepReced));
#assert System() reaches goal2;

#define goal3 ((msglist[1][0]==1)&&(msglist[2][0]==2)&& (msglist[3][0]==4));
#assert System() reaches goal3;

#define goal4 (((msglist[1][0]==1)&&(msglist[2][0])==2&&(msglist[3][0]==4))&&
               ((distancelist[msglist[1][0]][0]==1)&&
               (distancelist[msglist[2][0]][msglist[1][0]]==1)&&
               (distancelist[msglist[3][0]][msglist[2][0]]==1)));
#assert System() reaches goal4;

#define goal5 ((call(KeyValue,msgreply[1][1]))==KeyTj[1]&&
               (call(KeyValue,msgreply[0][1]))==KeyTj[0]&&
               (call(MACValue,KeySD,msgreply[1][0]))==MACTar);
#assert System() reaches goal5;
```

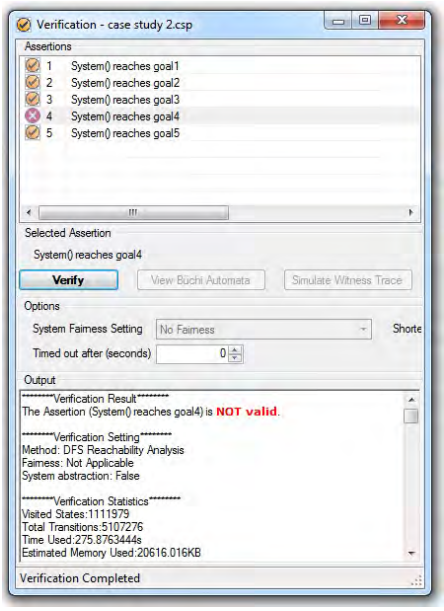


Fig. 6. The Result of Case II

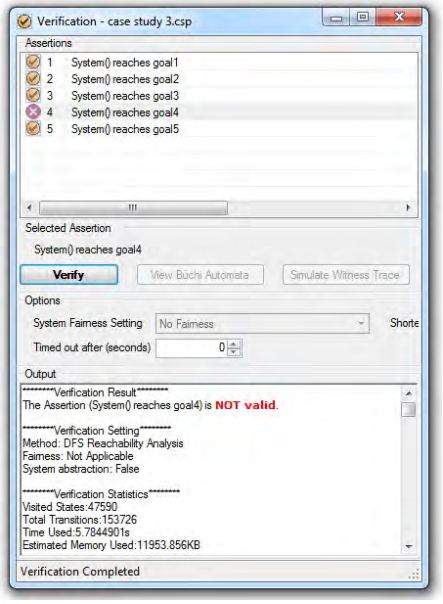


Fig. 7. The Result of Case III

Table 4. The Results of the Verification

	Property 1	Property 2	Property 3	Property 4	Property 5
Case Study 1	P	P	P	P	P
Case Study 2	P	P	P	F	P
Case Study 3	P	P	P	F	P

5.3. Results Analysis

The results of the verification are shown in Table 4. Here, P is the abbreviation of Pass which means the case study reaches the goal and passes the verification of the property. And F, shorted for Fail, means the case does not reach the goal. Through the table, we see that Case I reaches all of the five goals because of no fake path existing in Case I. Conversely, both Case II and Case III fail the verification of property 4 indicating there are fake pathes in these two case studies. Here, we discuss the details about these two results.

In the Case II, when the intruder C receives the message $\langle msg_{req}, h_A, (A), (M_A)\rangle$ from A, it will wait until the message $\langle msg_{req}, h_E, (A, B, E), (M_A, M_B, M_E)\rangle$ from E arrives. Then C will change the message especially the node list as (A, B, C) to D. The modified message is $\langle msg_{req}, h_C, (A, B, C), (M_A, M_B, M_C)\rangle$, where $h_C = H(C, H(B, h_A))$ and M_B can be deduced from the known information. After C gets the reply from D, it will cache the reply message until its own TESLA key and the TESLA key of node B have been released, then it will send the reply to A directly. In fact, (S, A, B, C) is a fake path that S does not

recognize. Meanwhile, we consider the Active-1-2 intruder in Case III. When the intruder C receives the request $\langle msg_{req}, h_A, (A), (M_A)\rangle$ from A, it saves the message into his private list, which the other corrupted node can also access. Then, the intruder whose identifier is also C will receive the message from B, it will change the node list as (A, C) and send it to D. After C receives the reply message from D, it will also put the reply message into its own list, then the other node whose identity is also named C will send the message back to A ignoring the intermediate node B. We also give the User Interface Figure, Figure 6 and Figure 7, which show the result of the verification of Case II and Case III in PAT.

6. Discussion

In Mobile Ad Hoc Networks, typical routing protocols are divided into three types, such as the proactive routing protocols, the on-demand (or reactive) routing protocols and the mixed routing protocols. In our paper, we focus on the on-demand routing ones, i.e. AODV [2,25], DSR [11], Ariadne, etc., in which the initial node will begin the process of route discovery to the destination only when it has a data packet needed to be sent to the destination. Through the comprehensive comparison and analysis, we find that all the on-demand routing protocols have the phase of the route discovery and we think that the modeling approach presented in our paper is also applicable to other protocols, which are based on the on-demand routing protocols and have the route discovery process.

The route discovery includes two processes: broadcasting the request message and unicasting the reply message. We can abstract the process of the route discovery of all the on-demand routing protocols as shown in Figure 8 below:

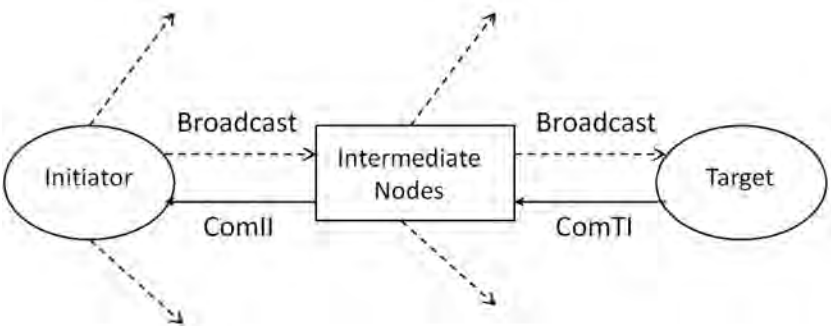


Fig. 8. The Abstract Model Diagram of the Route Discovery Process

We can define sets, i.e., **Initiator**, **Node**, **Target**, **MSG**, and channels such as **Broadcast**, **ComII** and **ComTI**, as mentioned in the previous sections. Some

more sets and channels can be added as needed. We give a general formalization here:

$$
\begin{aligned}
&Initiator(parameter) =_{df} |||_{X_i \in Node \wedge i \in [1,n] \wedge n \in N}(\\
&\quad Broadcast!msg_{req}....Initiator.X_i \rightarrow \\
&\qquad \text{/*More processes can be added here*/} \\
&\quad ComII?msg_{rep}....X_i.Initiator) \rightarrow Initiator(parameter)
\end{aligned}
$$

In the whole discovery process, the initiator will broadcast the request message to all of the nodes only one hop away from it, and receive the reply message from some intermediate node or the target. The actions of the intermediate nodes are more complex, that are receiving the request and the reply messages, broadcasting the request message and unicasting the reply message. Here, we do not discuss the details between different intermediate nodes, considering all the intermediate nodes as a whole part, shown below:

$$
\begin{aligned}
&Node(parameter) =_{df} (|||_{X_n \in Node \wedge n \in N}(\\
&\quad Broadcast?msg_{req}....Initiator.X_n \rightarrow \\
&\qquad \text{/*More processes can be added here*/} \\
&\quad Broadcast!msg_{req}....X_n.Target \rightarrow \\
&\qquad \text{/*More processes can be added here*/} \\
&\quad ComTI?msg_{rep}....Target.X_n \rightarrow \\
&\qquad \text{/*More processes can be added here*/} \\
&\quad ComII!msg_{rep}....X_n.Initiator) \rightarrow Node(parameter)
\end{aligned}
$$

The target node will receive the request message and unicast the reply message to the intermediate nodes. It can be formalized as follows:

$$
\begin{aligned}
&Target(parameter) =_{df} \\
&\quad Broadcast?msg_{req}....X_n.Target \rightarrow \\
&\qquad \text{/*More processes can be added here*/} \\
&\quad ComTI!msg_{rep}....Target.X_n \rightarrow Target(parameter)
\end{aligned}
$$

As mentioned above, we give a general framework to formalize the communication entities as CSP processes of route discovery of the on-demand routing protocols. Some other models, such as intruder model, clock model, etc., can be modeled as needed and the corresponding processes can also be added into our general framework. For instance, in our paper, we focus on the Ariadne protocol, which is an efficient and well-known on-demand secure protocol of MANETs. It mainly concerns about how to prevent a malicious node from compromising the route using three authentication mechanisms. Therefore, we add the security authentication mechanisms into our model and we also define some other processes to describe the security mechanisms.

7. Conclusion and Future Work

In this paper, we proposed a CSP model of the Ariadne secure route protocol. All the communication entities of the protocol, involving the initiator node, the intermediate nodes, and the target, have been abstracted as processes respectively. They share the same clock to realize the effect of asymmetric key encryption through clock synchronization and detention. Besides, we also proposed an intruder model in which the intruder can perform the attacks. We also discuss that our modeling approach is applicable to other on-demand routing protocols, which have the route discovery process. Furthermore, we use PAT, a model checker to implement and validate our formal model automatically. Three case studies are given and we verify five properties: Deadlock Freedom, Message Consistency, Node List Security, Fake Path Nonexistence and End-to-End Nodes Authentication. The verification result demonstrates that some fake paths indeed exist in the Ariadne protocol.

In the future, we would work on the formalizing of the other phase of the Ariadne protocol, including the process of the route maintain. Besides, based on the whole achieved model of the Ariadne protocol, further verification by using PAT is also an interesting topic to be explored. We also want to propose a research methodology to study the discovery process of the on-demand routing protocols for ad hoc networks.

Acknowledgments. The authors gratefully acknowledge support from the Danish National Research Foundation and the National Natural Science Foundation of China (Grant No. 61061130541) for the Danish-Chinese Center for Cyber Physical Systems. This work was also supported by National Basic Research Program of China (No. 2011CB302904), National High Technology Research and Development Program of China (No. 2011AA01 0101 and No. 2012AA011205), National Natural Science Foundation of China (No. 61021 004), and Shanghai Leading Academic Discipline Project (No. B412).

References

1. Ács, G., Buttyán, L., Vajda, I.: Provably Secure On-Demand Source Routing in Mobile Ad Hoc Networks. IEEE Trans. Mob. Comput. 5(11), 1533–1546 (2006)
2. Belding-Royer, E.M., Perkins, C.E.: Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Protocol. In: MOBICOM. pp. 207–218 (1999)
3. Bergstra, J.A., Klop, J.W.: Algebra of Communicating Processes with Abstraction. Theor. Comput. Sci. 37, 77–121 (1985)
4. Brookes, S.D., Hoare, C.A.R., Roscoe, A.W.: A Theory of Communicating Sequential Processes. J. ACM 31(3), 560–599 (1984)
5. Buttyán, L., Vajda, I.: Towards provable security for ad hoc routing protocols. In: Proc. 2nd ACM workshop on Security of ad hoc and sensor networks. pp. 94–105. SASN '04, ACM, New York, NY, USA (2004)
6. Chen, C., Dong, J.S., Sun, J., Martin, A.: A verification system for interval-based specification languages. ACM Trans. Softw. Eng. Methodol. 19(4) (2010)

7. Ding, J., Zhu, H., Li, Q.: Formal Specification of Automatic DMARF Based on CSP. In: Engineering of Autonomic and Autonomous Systems (EASe), 2011 8th IEEE International Conference and Workshops on. pp. 32 –39 (4 2011)
8. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall (1985)
9. Hu, Y.C., Johnson, D.B., Perrig, A.: SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks. Ad Hoc Networks 1(1), 175–192 (2003)
10. Hu, Y.C., Perrig, A., Johnson, D.B.: Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. Wireless Networks 11(1-2), 21–38 (2005)
11. Johnson, D.B., Maltz, D.A.: Dynamic Source Routing in Ad Hoc Wireless Networks. Mobile Computing pp. 153–181 (1996)
12. Jordan, R., Abdallah, C.: Wireless communications and networking: an overview. Antennas and Propagation Magazine, IEEE 44(1), 185 –193 (2 2002)
13. Lin, C.H., Lai, W.S., Huang, Y.L., Chou, M.C.: Secure Routing Protocol with Malicious Nodes Detection for Ad Hoc Networks. Advanced Information Networking and Applications Workshops, International Conference on 0, 1272–1277 (2008)
14. Liu, Y., Sun, J., Dong, J.S.: An Analyzer for Extended Compositional Process Algebras. In: ICSE Companion. pp. 919–920. ACM (2008)
15. Liu, Y., Sun, J., Dong, J.S.: Analyzing Hierarchical Complex Real-time Systems. In: FSE 2010. pp. 365–366 (2010)
16. Liu, Y., Sun, J., Dong, J.S.: PAT 3: An Extensible Architecture for Building Multi-domain Model Checkers. In: ISSRE. pp. 190–199 (2011)
17. Liu, Z.: The Security Analysis of Routing Protocol for Ad Hoc Networks. Journal of Huangshi Institute of Technology 23(4), 29–33 (8 2007)
18. Lowe, G., Davies, J.: Using CSP to Verify Sequential Consistency. Distributed Computing 12(2-3), 91–103 (1999)
19. Lowe, G., Roscoe, A.W.: Using CSP to Detect Errors in the TMN Protocol. IEEE Trans. Software Eng. 23(10), 659–669 (1997)
20. Luu, A.T., Sun, J., Liu, Y., Dong, J.S., Li, X., Tho, Q.T.: SeVe: automatic tool for verification of security protocols. Frontiers of Computer Science in China 6(1), 57–75 (2012)
21. Luu, A.T., Sun, J., Liu, Y., Dong, J.S., Li, X., Tho, Q.T.: SeVe: automatic tool for verification of security protocols. Frontiers of Computer Science in China 6(1), 57–75 (2012)
22. Mauve, M., Widmer, A., Hartenstein, H.: A survey on position-based routing in mobile ad hoc networks. Network, IEEE 15(6), 30 –39 (Nov/Dec 2001)
23. Mazur, T., Lowe, G.: Counter Abstraction in the CSP/FDR setting. Electr. Notes Theor. Comput. Sci. 250(1), 171–186 (2009)
24. Milner, R.: A Calculus of Communicating Systems, Lecture Notes in Computer Science, vol. 92. Springer (1980)
25. Perkins, C.E., Belding-Royer, E.M.: Ad-hoc On-Demand Distance Vector Routing. In: WMCSA. pp. 90–100 (1999)
26. Perrig, A., Canetti, R., Song, D.X., Tygar, J.D.: Efficient and Secure Source Authentication for Multicast. In: Proc. the Network and Distributed System Security Symposium. The Internet Society (2001)
27. Perrig, A., Canetti, R., Tygar, J.D., Song, D.X.: Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In: IEEE Symposium on Security and Privacy. pp. 56–73 (2000)
28. Pura, M.L., Bica, I., Patriciu, V.V.: On modeling and formally verifying secure explicit on-demand ad hoc routing protocols. In: Proc. 2nd International Conference on Software Technology and Engineering. vol. 2, pp. 215–220 (10 2010)

29. Rohrmair, G.T., Lowe, G.: Using CSP to Detect Insertion and Evasion Possibilities within the Intrusion Detection Area. In: Proc. 1st International Conference on Formal Aspects of Security. pp. 205–220. Springer (2002)
30. Roscoe, A.W.: The theory and practice of concurrency. Prentice Hall (1998), http://www.cs.ox.ac.uk/people/bill.roscoe/publications/68b.pdf
31. Roscoe, A.: Understanding Concurrent Systems. Springer (2010), http://www.comlab.ox.ac.uk/ucs
32. Sanzgiri, K., Dahill, B., Levine, B.N., Shields, C., Belding-Royer, E.M.: A Secure Routing Protocol for Ad Hoc Networks. In: Proc. 10th IEEE International Conference on Network Protocols. pp. 78–89. IEEE Computer Society (2002)
33. Sivakumar, K.A., Ramkumar, M.: Improving the resiliency of Ariadne. In: Proc. 9th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks. pp. 1–6. IEEE (June 2008)
34. Sun, J., Liu, Y., Dong, J.S.: Model Checking CSP Revisited: Introducing a Process Analysis Toolkit. In: Proc. 3rd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation. pp. 307–322. Springer (2008)
35. Sun, J., Liu, Y., Dong, J.S., Pang, J.: PAT: Towards Flexible Verification under Fairness. Lecture Notes in Computer Science, vol. 5643, pp. 709–714. Springer (2009)
36. Sun, J., Liu, Y., Dong, J.S., Pu, G., Tan, T.H.: Model-based Methods for Linking Web Service Choreography and Orchestration. In: APSEC 2010. pp. 166 – 175 (2010)
37. Sun, J., Liu, Y., Song, S., Dong, J.S., Li, X.: PRTS: An Approach for Model Checking Probabilistic Real-Time Hierarchical Systems. In: Qin, S., Qiu, Z. (eds.) Formal Methods and Software Engineering. Lecture Notes in Computer Science, vol. 6991, pp. 147–162. Springer Berlin / Heidelberg (2011)
38. Sun, J., Song, S., Liu, Y.: Model Checking Hierarchical Probabilistic Systems. In: Dong, J.S., Zhu, H. (eds.) Formal Methods and Software Engineering - 12th International Conference on Formal Engineering Methods, ICFEM 2010, Shanghai, China, November 17-19, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6447, pp. 388–403. Springer (2010)
39. Suresh, S., Mike, W., S., R.C.: Power-aware routing in mobile ad hoc networks. In: Proc. 4th annual ACM/IEEE international conference on Mobile computing and networking. pp. 181–190. MobiCom '98, ACM, New York, NY, USA (1998)
40. Tuan, L.A.: Modeling and Verifying Security Protocols Using PAT Approach. Secure Software Integration and Reliability Improvement Companion, IEEE International Conference on 0, 157–164 (2010)
41. Wang, M., Zhu, H., Zhao, Y., Liu, S.: Modeling and Analyzing the (mu)TESLA Protocol Using CSP. In: TASE. pp. 247–250 (2011)
42. Wu, X., Liu, S., Zhu, H., Zhao, Y., Chen, L.: Modeling and Verifying the Ariadne Protocol Using CSP. In: ECBS. pp. 24–32 (2012)
43. Zapata, M.G., Asokan, N.: Securing ad hoc routing protocols. In: Proc. 2002 ACM Workshop on Wireless Security. pp. 1–10. ACM (2002)
44. Zheng, M., Sun, J., Liu, Y., Dong, J.S., Gu, Y.: Towards a Model Checker for NesC and Wireless Sensor Networks. In: Qin, S., Qiu, Z. (eds.) Formal Methods and Software Engineering. Lecture Notes in Computer Science, vol. 6991, pp. 372–387. Springer Berlin / Heidelberg (2011)
45. Zheng, M., Sun, J., Sanán, D., Liu, Y., Dong, J.S., Gu, Y.: Towards bug-free implementation for wireless sensor networks. In: SenSys. pp. 407–408 (2011)

Xi Wu received her BSc in Software Engineering from Software Engineering Institute, East China Normal University in 2011. She is currently a master student in Formal Methods with the same institute. Her research interests include process algebra and its applications, program analysis and verification, and web services.

Huibiao Zhu is Professor of Computer Science at Software Engineering Institute, East China Normal University. He received his BSc in Mathematics and MSc in Computer Science in 1989 and 1992 respectively, all from East China Normal University. He earned his PhD in Formal Methods from London South Bank University in 2005. His research interests include the following areas: (1) semantics theory, including process algebra and its applications; (2) unifying theories of programming; (3) formal design, specification and verification in hybrid systems.

Yongxin Zhao held a PhD degree in Formal Methods from Software Engineering Institute, East China Normal University in 2012. He is currently a research fellow at School of Computing of National University of Singapore. His research interests include program analysis and verification, semantics theory, web services. He owns more than 15 referred publications.

Zheng Wang received his BSc and PhD from Software Engineering Institute, East China Normal University in 2007 and 2012 respectively. Now he is a software requirement engineer in the Software Development Department at Beijing Institute of Control Engineering, China Academy of Space Technology. His main research topic focuses on the automatization and formalization of requirement analysis for embedded control software.

Si Liu received his BSc and MSc from Software Engineering Institute, East China Normal University in 2009 and 2012 respectively. He is currently a PhD student with the Formal Methods and Declarative Languages Laboratory, the Department of Computer Science, University of Illinois at Urbana-Champaign. His research interests are in the areas of formal methods and programming languages.

Received: June 1, 2012; Accepted: August 30, 2012.

DOI:10.2298/CSIS120531010M

System Design for Passive Human Detection using Principal Components of the Signal Strength Space

Bojan Mrazovac[1], Milan Z. Bjelica[1], Dragan Kukolj[1], Branislav M. Todorović[2] and Saša Vukosavljev[2]

[1] Faculty of Technical Sciences, Trg Dositeja Obradovića 6,
21000 Novi Sad, Serbia
{bojan.mrazovac, milan.bjelica, dragan.kukolj}@rt-rk.com
[2] RT-RK, Institute for Computer Based Systems, Narodnog Fronta 23a,
21000 Novi Sad, Serbia
{ branislav.todorovic, sasa.vukosavljev}@rt-rk.com

Abstract. In this article, device-free human presence detection method based on principal components analysis of the radio signal strength variations is proposed. The method increases user awareness for automated power management interaction in residential power saving systems. Motivation comes from a need for decreasing the installation complexity and development costs induced by the integration of specific human presence detection sensors. The method exploits the fact that a human body interferes with radio signals by introducing irregularities in the radio signature which indicate possible human presence. By analyzing radio signals between radio transceivers embedded in 2.4 GHz wireless power outlets, the original environment is not visually modified and a certain level of sensorial intelligence is introduced without additional sensors. The analysis of the signal strength variations in principal components space enhances the detection accuracy level of human presence detection method, retaining low installation costs and improving overall energy conservation.

Keywords: energy awareness, human presence detection, principal components analysis, radio irregularity, RSSI, smart outlets, Zigbee.

1. Introduction

Due to the rise of the global energy demands, the electricity price increase and the limitation of natural resources used for electricity generation, several considerations about the energy saving have been brought up recently [1], [2], [3], [4], [5]. An optimized approach for residential electric energy conservation requires installation of power metering devices. The European Union and the European Regulators' Group for Electricity and Gas have proposed an initiative [6] to encourage the installation of smart power meters

in all homes across Europe during the next decade. The most frequently used solution for smart power metering is made in a form of smart power outlets. Smart outlets offer the possibility for additional energy-related services such as on-demand power management, overview of the consumed energy and power switching of plugged devices. Such an approach provides more accessible information which help people to use energy more efficiently.

Although consumers are aware about their power consumption, their habits are very difficult to change and in many cases no corrective actions that would decrease the power consumption are taken. Therefore, there exists a need for automated power management solution, which does not require a user to intervene. To enable the automatic response, it is necessary to establish the interaction with the environment by integrating a number of sensors, mainly for human presence detection. An example of interactive energy saving platform is proposed in the previous work as "*Ecosystem for Smart Home*" (ESH) [5]. The ESH improves the power consumption efficiency by connecting smart power outlets and smart light switches, which are part of pre-existing electrical installations, with human presence detection sensors. The integration of sensors and smart power nodes increases user awareness of the smart home for the advanced automated power management.

Human presence detection method, proposed in this article, is motivated by a need for decreasing the installation complexity and development costs induced by the integration of specific sensors in smart energy environment. As opposed to the original ESH framework which incorporates various human presence detection sensors, the proposed method detects human presence without specific sensors. The detection is enabled only by analyzing and quantifying radio signal strength variations at the inputs of radio transceivers embedded in wireless nodes. This approach exploits the fact that human bodies interfere with radio signals, causing fading and shadowing effects. Therefore, irregularities in the radio signature, given in a form of received signal strength indicator's (RSSI) variations, are considered as an indication of possible presence in the room. The method extracts principal components from a covariance matrix composed of samples that present signal strengths gathered from wireless links inside a room. Principal component analysis enhances the accuracy level with small percentage of false alarms and improves the overall probability of human presence detection. Since the most of indoor environments contain power outlets, replacing them with smart power outlets would not modify the environment visually, but existing electrical installations would be extended with the detection capability. The use of radio irregularity from radio links in an already installed network of wireless power outlets preserves the transparency of smart home devices, supports high level of sensorial intelligence and has low installation cost.

The paper is structured as follows. In Section 2, an overview of device-free methods for human presence detection is given, including theoretical background. In addition, an overview of smart energy systems for residential use is introduced. The proposed human presence detection method is explained in Section 3. The ESH system which incorporates the proposed

method is described in Section 4. Experimental results are given in Section 5. At the end of the paper, in Section 6, a conclusion with an idea for the future improvement is given.

2. Related Work and Theoretical Framework

Radio irregularity is a common phenomenon which is often considered as a shortcoming of radio networks. A number of experiments set in [7] and [8] explain that radio irregularity is mainly caused by two factors: device properties and the propagation medium. Device properties include: the antenna type, the transmitter's radiated power, the receiver's sensitivity, and signal-to-noise ratio. Medium properties include the background noise and the environmental factors such as obstacles within the propagation path. When the signal travels through a medium, it may be absorbed, scattered, reflected or diffracted. At microwave frequencies, absorption by molecular resonance is a major factor affecting the radio propagation [9]. Scattering occurs when the signal propagates through a medium which contains a large number of objects smaller than the signal's wavelength. Reflection occurs when the signal encounters an object which is larger than the signal's wavelength. Diffraction occurs when the signal encounters an irregular surface, such as sharp edges.

The irregularity of the radio signals is even more expressed when a human body encounters the signal in its propagation path. The human body is comprised of molecules of water which are able to additionally absorb, diffract, scatter or reflect the energy of the radio signal. Therefore, the presence of a human within the wireless network range results in significant signal strength variations at the receiver, whereas the degree of the signal strength variation is correlated with the level of human motion.

Woyach *et al.* [10] report that the shadowing effect caused by a human subject moving in the line-of-sight path between two communicating wireless nodes can be used for human motion detection. Such an approach, mainly based on RSSI variations analysis is extended for the outdoor people counting mechanism [11]. Lee *et al.* [12] investigated the feasibility of intrusion detection by characterizing the signal strength fluctuations and translating them into sufficient information that corresponds to an intruder's activity. The presented idea is extended for an indoor automated people counting mechanism [13]. Intruder detection method [14] enabled by exploiting RSSI considerations, confirms the hypothesis that irregularities in the RSSI signature can be used as human presence indication. Through distributed processing of RSSI samples, nodes deployed in an indoor environment can also detect human presence and possibly help in localizing and tracking moving individuals, as shown by Kaltiokallio *et al.* [15]. The use of RSSI variations due to radio irregularity for security threats detection alongside a roadway, explained by Puzo *et al.* [16], demonstrates the ability of passive wireless sensor networks (PWSN) to be applied for the outdoor

surveillance. Patwari and Wilson [17] explain how multi-path fading can be used for the benefit of device-free localization systems. In such environments denoted as “RF sensor networks”, a human position can be inferred by measuring the absorption, reflection, scattering and diffraction of an electromagnetic wave, intersected by the human body. Device-free human localization in indoor environments using “RF sensor networks” is also the topic of the research presented by Deak *et al.* [18]. The phrase “RF sensor network” comes from the fact that the wireless network itself is the sensor, using RF signals to probe the environment. It is important to mention that a human does not need to be carrying a wireless device to be detected. Zhang *et al.* [19] proposed an RF sensor network operating at 870 MHz for indoor people tracking. The positioning method is based on capturing the RSSI dynamics of the reflected signals, which varies due to subject movement. That approach is further extended in [20] to implement the system capable of multiple persons tracking, simultaneously moving in the monitored area. The algorithm is based on distributed dynamic clustering that improves the localization accuracy when multiple subjects are present. Moussa and Youssef [21] demonstrate the feasibility of device-free passive intruder detection and localization by using the moving average of RSSI variance to detect the intrusion events.

The most of the existing residential smart energy solutions have one important attribute in common: they rely on various sensor technologies and sensor networks, such as [1], [2], [5], [22], [23], [24], [25], [26]. Because of the important impact of sensor networks applications in smart home’s environmental challenges, the authors of this paper have tried to make a synthesis between “RF sensor networks” and residential smart energy systems. In order to detect human presence in smart energy infrastructure, an algorithm that characterizes the signal strength variations, has been previously proposed in [27] and [28]. The algorithm is incorporated into the smart power outlets, by enabling them to detect human presence only by analyzing and quantifying radio signal strength variations incorporated in exchanged messages. The RSSI standard deviation and discrepancies between the mean value of a set of RSSI samples and the set’s min and max values are compared to define the interval of the initial signal strength variation. During the runtime, each outlet is polled periodically by the specific controller device, to obtain their current RSSI values from the messages exchanged with other outlets. The algorithm compares read RSSI values with the interval’s bounds. When the human steps into the monitoring area, the signal strength variation exceeds the previously set bounds and reports the presence of a subject. The shortcoming of such an approach is that the algorithm monitors RSSI variation intensity on each link independently. It is enough that the interval is exceeded only at one link and the detection will be reported. This is also the case for many related researches that were performed in a controlled environment. Unfortunately, in real environment, the external noise (e.g. interferences from another room, or single link variations for specific positions in the room) can disturb a radio link in the monitoring room, resulting in reported false alarms.

In order to improve the presence detection for real-world applications, the RSSI processing algorithm resistant to external noise is proposed in this paper. To meet the requirement, the authors propose the use of Principal Components Analysis (PCA). The RSSI variation intensity is given as a function over the entire network of radio links (RSSI) in the monitoring room. The links are simultaneously processed, therefore in a case when a few links are interfered with the external noise, the power of the majority of links will minimize, or even entirely suppress the noise. PCA successfully filters out the disturbed signals in order to preserve the correct detection.

3. Human Presence Detection Method based on Principal Components Analysis of the Signal Strength

Principal components analysis [29], [30] is a useful statistical technique used in many forms of statistical analysis, from biomedical signal processing [31] to computer graphics and pattern recognition [32]. It presents a simple, non-parametric method for extracting relevant information from confusing and large data sets. PCA is a variable reduction procedure. It is useful when samples are obtained on a large number of variables that are mutually correlated. PCA helps identifying patterns in the data, and expressing the data in a way that highlights their similarities or differences. Because of this variables redundancy, it is possible to reduce the large set of observed variables into a smaller number of principal components while retaining as much as possible of the variation present in the original data set. As the final result, each principal component contains new information about the original data and is ordered so that the first few components account for most of the variability. In the proposed algorithm, PCA compresses raw RSSI inputs obtained from each radio link, in order to extract principal components that are used to emphasize the variability of the signal strength.

A set of RSSI samples obtained from a communication link between two wireless nodes (outlets) inside the same room forms the zero-mean column vector $linkToNod_k$:

$$linkToNod_k = \begin{bmatrix} sample(1) \\ sample(2) \\ \dots \\ sample(\mathrm{N}) \end{bmatrix}. \quad (1)$$

Vector $linkToNod_k$ stores the information about RSSI signature from the link between a node which is currently polled by the home controller, and another node which communicates with the polled node. Each value *sample(i)* denotes an RSSI sample obtained from that link, whereas the counter *i* takes its values from 1 to *N*, for *N* that is the number of samples in the observed time window. Counter *k* takes its values from 1 to *K-1*, where *K*

represents the number of all active nodes inside the detection scope. Links toward remaining nodes represent an ensemble of *K-1* sensing links. The entire ensemble can be compactly expressed by the *N×(K-1)* data matrix *Nod*, which defines *(K-1)* observations of the random process:

$$Nod = \left[linkToNod_1 \quad linkToNod_2 \quad . \quad linkToNod_{k-1} \right] \tag{2}$$

Once the samples are collected, the shift interval which includes number of *p* samples from each column of matrix *Nod* needs to be defined. Over new set of *p* samples, the standard deviation (STD) is calculated. The interval of first *p* samples is further represented through the value $stdLink_p$, which shows the standard deviation of the vector *p* calculated over a single link. Afterwards, the STD is performed for the rest of the columns of matrix *Nod* which contain the data from other active links. That way, the shifting interval and the STD calculation applied to *p* samples from each column can be saved in a new vector z_k :

$$z_k = \left[stdLink_1 \quad stdLink_2 \quad ... \quad stdLink_{K-1} \right] . \tag{3}$$

The procedure for the creation of the vector z_k is repeated for each *K*-th node, after the node is polled by the home controller. Counter *k* denotes the id of the polled node and takes its values from 1 to *K*, where *K* is the number of nodes in the environment. The calculated values are stored into new *(K-1)×K* matrix *X*:

$$X = \left[z_1 \quad z_2 \quad ... \quad z_K \right] , \tag{4}$$

whose columns represent transposed vectors z_k, for each wireless node. After the samples are collected it is important to determine how much the dimensions vary from the mean value with respect to each other. For that purpose the statistical measure covariance (*cov*) is used:

$$\text{cov}(z_i, z_j) = \frac{\sum_{s=1}^{K-1} (z_i(s) - \overline{z_i})(z_j(s) - \overline{z_j})}{((K-1)-1)} , \tag{5}$$

$$i, j = [1, K] \ \wedge \ i \neq j ,$$

where $\overline{z_i}$ and $\overline{z_j}$ denote mean values from the set of samples per vectors z_i and z_j, respectively:

$$\overline{z_i} = \frac{1}{K-1} \sum_{s=1}^{K-1} z_i(s), \ \overline{z_j} = \frac{1}{K-1} \sum_{s=1}^{K-1} z_j(s) . \tag{6}$$

The expression (5) is divided by *(K-1)-1*, because the data represent only a sample. This gives the result that is closer to the standard deviation, which would result if the entire population is used. As the next step of the algorithm, all the possible covariance values between the variables should be calculated

and stored into covariance matrix C_X. By using the equation (4) that defines the matrix of links *X*, the covariance matrix can be expressed as:

$$C_X = \frac{1}{(K-1)-1} XX^T \ . \tag{7}$$

Each row of *X* corresponds to all measurements of a particular link. Each column of *X* corresponds to a set of measurements from one particular polling cycle. The matrix C_X captures correlations between all the possible pairs of measurements. A large value of C_X indicates high redundancy between measurements, whereas small indicates low redundancy.

PCA enables the linear transformation that maps the data from a higher dimensional space to a lower dimensional space. Low dimensional space is determined by the strongest eigenvectors of the covariance matrix C_X, known as principal components. The eigenvectors of C_X are non-zero vectors that, after being multiplied by the matrix C_X remain proportional to the original vector or become zero. An eigenvalue represents the scalar which defines how the eigenvector changes (stretches, flips, shrinks or leaves unchanged) when it gets multiplied by matrix C_X.

If *W* is a vector space and *w* is a vector from that space, then *w* represents an eigenvector of matrix C_X with eigenvalue λ, defined as:

$$C_X w = \lambda w \ . \tag{8}$$

The eigenvalues of C_X can be calculated as the roots of characteristic polynomial which can be derived from the expression:

$$\det(C_X - \lambda I) = 0 \ , \tag{9}$$

where *det* stands for determinant and *I* is the *K*×*K* identity matrix. The eigenvectors correspond to principal components whereas the eigenvalues correspond to the variance defined by the principal components. Once the eigenvectors are found from the covariance matrix C_X, the next step is to order them by eigenvalues, highest to lowest, which orders the principal components by their significance. By ignoring less significant components, the final data set will have less dimensions than the original. The last step of the algorithm is to form the *Feature Vector fv* which is constructed by using the most significant eigenvalues. By analyzing the eigenvalues saved in the vector *fv*, the presence of a human can be determined. No presence implicates very low RSSI variations and therefore low eigenvalues (very close to value 0). When a human subject is present, RSSI variations from wireless links are becoming higher, with strongly expressed deviations from the mean value, which implicates higher eigenvalues. The detection bound is set to be the maximal value from the *fv* during the phase of training (no humans in the room). During the runtime, the eigenvalues which are higher

than the bound, report human presence. Lower eigenvalues report the empty room.

4. Case Study – System Design for Residential Energy Awareness

In one of the previous papers a smart energy system for the residential use has been presented [5]. The system is comprised of the home controller device, 2.4GHz (*IEEE 802.15.4*) wireless smart outlets, 2.4GHz smart light switches and a number of residential sensors. All these devices are connected to the residential smart power network. By interpreting user-defined power saving schemes given in a form of XML based scripts [33] the user awareness of the entire system is increased. Increased awareness enables automation of instructions that generate ambient intelligence environment. The concept is depicted in Fig.1.

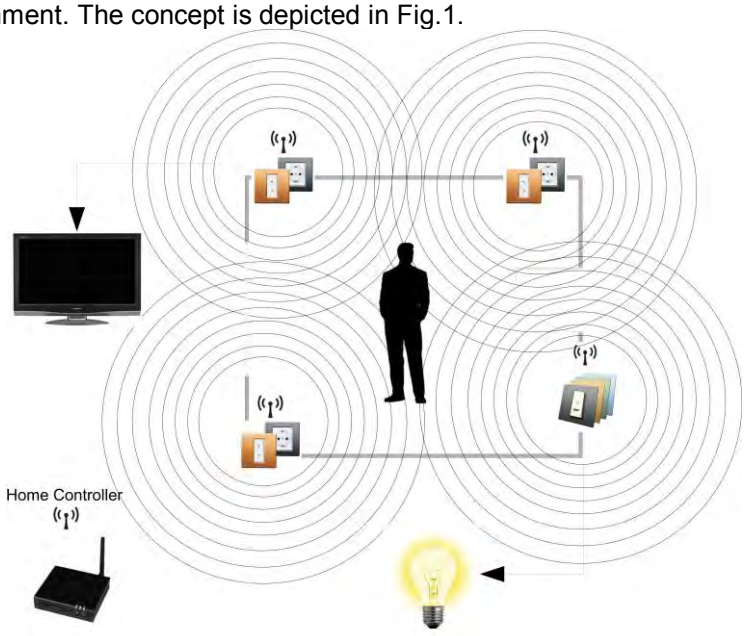


Fig. 1. The concept of human presence detection method based on wireless smart outlets, light switches and the analysis of RSSI variations

Implementation of the proposed method for presence detection requires at least two smart wireless power outlets, which can be combined with smart light switches. The communication control, periodic polling mechanism and the RSSI data analysis are implemented within the core software modules of the home controller. The home controller (illustrated in Fig. 2) is made in a form of a software platform based on POSIX/C open standards which provide

scalability. The software is platform independent and can be easily ported to various POSIX-based target controllers.

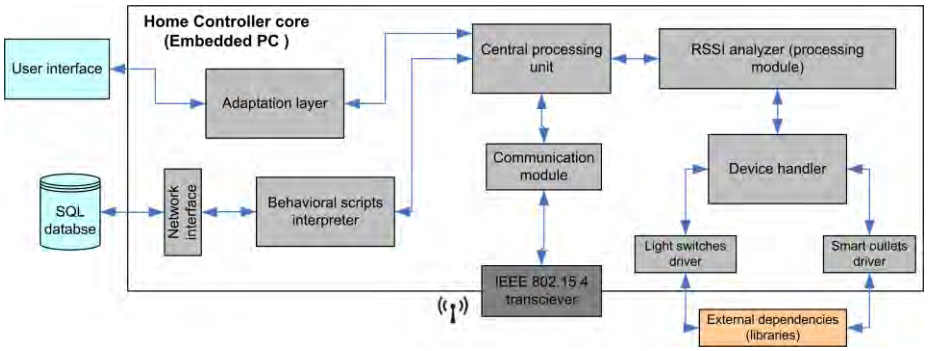


Fig. 2. The home controller software design. The home controller is comprised of adaptation layer which provides communication toward user interface; RSSI analyzer which polls each outlet and analyzes RSSI data; the communication module which provides communication with the smart wireless nodes and user interfaces; the device handler which provides device drivers for smart nodes; and the behavioral scripts interpreter which provides engine for user-defined power saving schemes execution

The device handler module connects device drivers for smart outlets and light switches with the central processing unit, providing a communication mechanism for wireless nodes polling and RSSI data reading. The device handler controls the message flow as the response on detection events. The RSSI analyzer enables periodic polling of wireless nodes to retrieve the current values of RSSI. The home controller polls each node (outlet) in turn on every 100ms and saves the received values in the local storage database. That way the system is able to detect even a human running with the fastest known speed without unnecessary frequent polling that can bring high processing loads to the system.

Once a node receives the polling command from the home controller it sends its RSSI table as a broadcasted message. The message contains a table of RSSI values toward all links (other wireless outlets) nearby. During the period of one node polling the other nodes are in the “listening” mode, so there is no interference or superposition of signals between them. The broadcasted message is received by the controller as well as by other nodes which update their RSSI tables with the values of signal strength received for that link. The nodes are able to receive the message from the controller as well as from neighboring nodes. Once the message is received, the RSSI analyzer saves the received values in the local database and waits for the next 100ms, to poll another node inside a room. After a polling cycle, the controller can generate a functional status by monitoring the principal components, extracted from the matrix of RSSI values, as explained in the previous section.

Smart outlets and light switches (shown in Fig. 3), presented in details in [5] and [34], fit into existing electrical installations, power sockets on the wall. Smart outlets provide power to electrical devices with standard flat, two-pole AC power plug (CEE 7/16) which is designed for voltages up to 250V and currents up to 2.5A. Besides simple on/off switching, sockets and light switches are able to pass any percentage of power to the consuming electric device (e.g. light dimmer). IEEE 802.15.4 transceiver (2.4GHz Zigbee) is used as the wireless communication module. Smart outlets are powered from 220-240Vac (±10%) 50Hz current electric power supply. It is an inexpensive and the safest way to provide full compatibility with the regulatory requirements. With an average current of 35mA and the operational voltage of 3.3V for an outlet and 2.4V for a switch, the power supply consumption is approx. 0.12W per an outlet and 0.08W per a switch.

Fig. 3. The smart power outlet and smart light switch; the retrofit design that fits into existing electrical installation on the wall (CEE 7/16 standard)

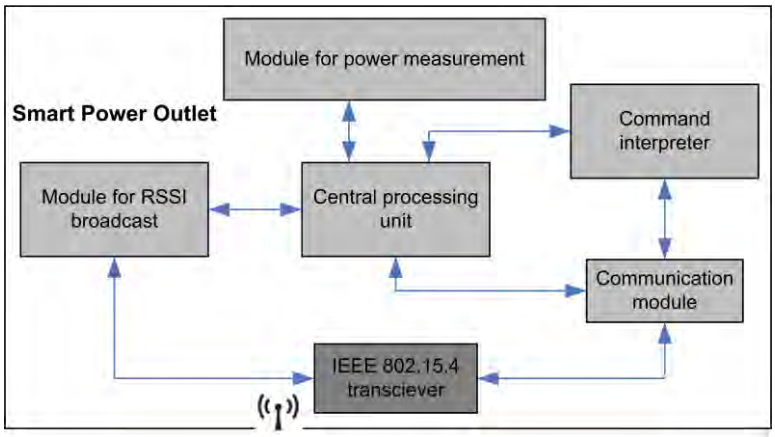


Fig. 4. The smart power outlet software design. Smart power outlet is comprised of: module for RSSI broadcast which sends a broadcast message on each polling instruction received from the home controller; the communication module which establishes Zigbee communication protocol with the controller and other smart nodes; the command interpreter which executes the switch and the dimming control; and the module for power measurement which provides consumption data

The smart outlets and light switches incorporate specific firmware which is implemented to enable: (1) the access to the consumption overview on

demand, (2) switching the plugged devices on or off, and (3) environmental sensing by broadcasting RF messages to neighboring nodes. The firmware modules are illustrated in Fig. 4.

The module for power measurement sends its current values periodically (each second) to the home controller. The power consumption for daily, weekly and monthly basis are processed within the home controller and stored into the local database. The module for RSSI broadcast waits for an event from the home controller which actuates the RSSI message broadcasting. The same module receives broadcasted messages from other nodes during the polling cycle. Parsed message is saved in the structure, which is provided to the home controller after the node is polled. Command interpreter executes commands received from the home controller, such as dimming control, switching the plugged device on or of, etc. The smart light switch firmware design is similar to the smart outlet firmware.

5. Experimental Results for Human Presence Detection

The test bed described in the previous section was installed in two buildings. In the first building, walls were made of concrete parts (exterior wall) and gypsum attached to the steel construction (interior wall) isolated with fiberglass wool. The gypsum wall thickness was 15cm and the concrete wall was 30cm. In the second building, the walls were made of aluminum and plastic covers, 30cm thick and mounted on steel construction, isolated with fiberglass wool. In each building a room was selected and four smart nodes were installed and placed strategically. Three of them (smart outlets) have been positioned at an elevation of 40cm above the floor and the last one (smart switch) was positioned at an elevation of 120cm above the floor. The testing room made of concrete and gypsum walls (further referred to as *R1*) was 536×530cm, whereas the room with aluminum and plastic walls (further referred to as *R2*) was 960×580cm large. The rooms' layouts and the positions of a subject (shown as points *P1-P5* for *R1*, apropos *P1-P6* for *R2*) and nodes positions (shown as squares *N1-N4*) are illustrated in Fig.5.

Coordinates of each node in *R1*, relatively to the central position, as well as positions of a testing subject, are given in Table 1. All the coordinates are given in cm, and measured relatively to the central position. The central position is located in the down left corner.

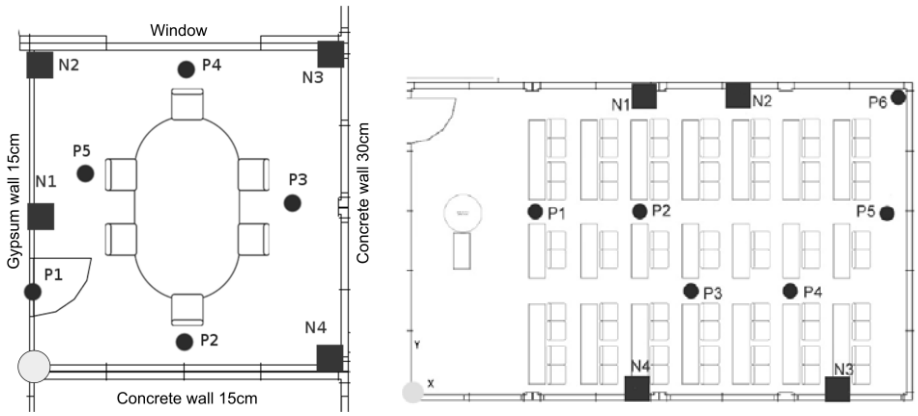


Fig. 5. The experimental rooms' layout. On the left - the room with concrete and gypsum walls; on the right - the room with aluminum walls with plastic slices

Table 1. Human subject's and wireless nodes' positions in R1

Node name	Node coordinates	Subject's position	Subject's coordinates
N1	(73, 211)	P1	(0, 78)
N2	(54, 477)	P2	(270, 75)
N3	(474, 428)	P3	(424, 254)
N4	(519, 66)	P4	(306, 420)
-	-	P5	(120, 255)

Coordinates of each node in *R2*, relatively to the central position, as well as positions of a testing subject are given in Table 2. The central position is also located in the down left corner. The coordinates are given in cm.

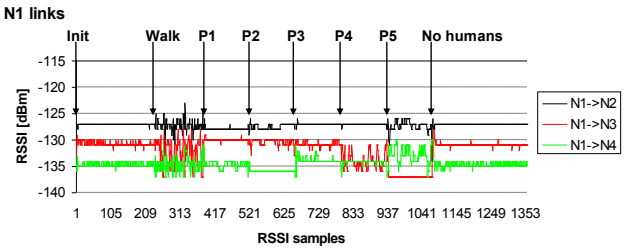
Table 2. Human subject's and wireless nodes' positions in R2

Node name	Node coordinates	Subject's position	Subject's coordinates
N1	(410,530)	P1	(220,305)
N2	(600,530)	P2	(410,305)
N3	(795,40)	P3	(500,150)
N4	(410,40)	P4	(690,150)
-	-	P5	(945,305)
-	-	P6	(955, 500)

According to scattering, diffraction, absorption and reflection of the signal in these environments, two test scenarios were defined. In the first scenario, the room was empty for a period of two minutes, and no detection was reported. Once a subject entered the room, he performed clockwise walking

around the table within the room, by passing the positions *P1-P5*, from the Fig. 5 - left. After one minute of walking, the subject was standing in each position *P1-P5*, for a minute, without movements. The scenario tried to confirm the hypothesis that the detection of human presence or movement is possible by analyzing the extracted principal components from the sets of RSSI variations retrieved from each wireless link. Sets of raw RSSI samples, before PCA processing, are logged and presented in Fig. 6. The values are given in dBm, but the idea is that the algorithm takes raw 8bit values into the processing. In that case no additional conversions are necessary during the runtime. The 8bit RSSI value is in signed 2's complement on a logarithmic scale with 1-dB step and must be corrected with an RSSI offset to get the real RSSI value in dBm. For CC2530 transceiver, the RSSI offset is 73 dB. Real RSSI is calculated by subtracting the RSSI offset from the converted 8bit RSSI value.

From the Fig. 6 it can be noticed that in the position *P1* the RSSI variation was emphasized only at the link *N1→N4* in both directions. It is explained as a result of signal reflection by the human body which was very close to the line-of-sight between outlets *N1* and *N4*. In the position *P2*, the human body shadowed the links *N1→N4* and *N4→N1*, and the most of the radio signal was absorbed by the human body which is the main reason for lower RSSI values. In the position *P2*, the links *N2→N4* and *N4→N2* were distorted with the reflection by the human body. Therefore, high RSSI variation in the position *P2* for links between outlets *N2* and *N4* can be noticed. Moreover, the position *P2* had slight influence to the links *N1→N3* and *N3→N1* that were distorted by the vicinity of human body which slightly reflected the signal. The human position *P3* mostly absorbed the signal from the links *N2→N4* and *N4→N2*, and reflected the signals from the links *N3→N4*, *N4→N3* and *N1→N4*, *N4→N1*. Position *P4* shadowed the links *N1→N3* and *N3→N1* and absorbed the signal. The position *P5* shadowed the links *N1→N3* and *N3→N1* and reflected the signals from the rest of links, except for *N3→N4* and *N4→N3* which were far from the current human position. At the end of the experiment the room was empty again for two minutes.



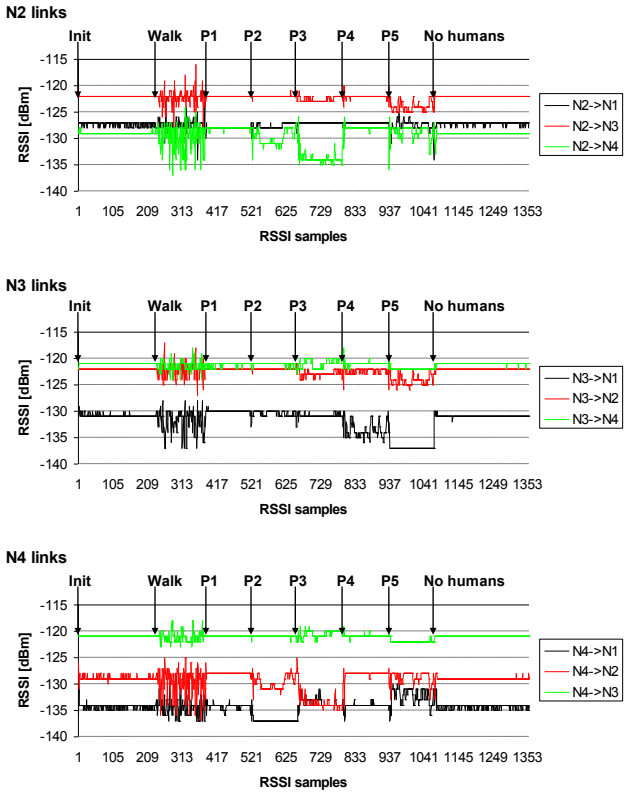


Fig. 6. Raw RSSI samples data observed in the first experimental room – R1

From the aspects of radio irregularity feature, the conclusion for the first scenario can be deduced: (1) RSSI for all wireless nodes that have communicated far from the human, varied slightly or had a constant value. (2) When the human was positioned closer to a node, without obstructing the line-of-sight, RSSI varied significantly. The larger variation is explained as the consequence of the signal reflection. (3) When the human obstructed the line-of-sight on one link, the RSSI did not vary much comparing to the other links, but was diverse comparing to the initial values.

It can be clearly noticed that the appearance of a human subject induced RSSI variations in the environment. Particularly, during subject's walking, RSSI variations have been emphasized at all links.

Once the data set is collected for a number of samples (interval *p* – the testing was performed for the interval of 12, 24 and 36 RSSI samples) the matrix *Nod* given by (2) is formed, for each link. After the matrix *X* is created and the matrix C_X is calculated, the principal components are extracted. The graphical presentations of principal components stored in the *Feature Vector fv* for the time interval of 550 seconds are shown in Fig. 7, Fig. 8 and Fig. 9,

depending of the interval *p* (12, 24 and 36, respectively). Different values for *p* are used to experimentally determine the optimal number of samples that can achieve accurate detection and preserve fast system response. The first 200 principal components are used to define the detection bound, and this phase is known as the training phase. During the training phase no human should be present in the room, otherwise high detection bound can be set. The detection bound is calculated as the maximal value of principal components from the training phase. The values higher than the calculated bound report detection.

In the Fig. 7 the result of the applied PCA for the shifting interval *p* which includes 12 samples is shown. The first high peak (after the sample 200) reports the presence of a human. As defined in the scenario, the human was walking for a minute (the following 200 samples). During the motion, the principal components powers are strongly emphasized and human presence can be easily detected with 100% accuracy. The human standing without movements is less expressed on principal components and the detection accuracy is lower. The following high peaks show the transitions from each position *P1-P5* to the next one. The defined human positions gradually affect the radio links in the room. All the links are not immediately obstructed with the human body and high discrepancies between them exist. Some of the links would still have low RSSI variations before their line-of-sight becomes intersected. As the human moves to the centre of the room (positions *P3*, *P4* and *P5*), the power of principal components increases. The overall detection accuracy, with *p* defined to include 12 samples, is approx. 75.3% for human presence (which includes walking and standing in each position *P1-P5*), whereas the accuracy for the empty room detection is 100%.

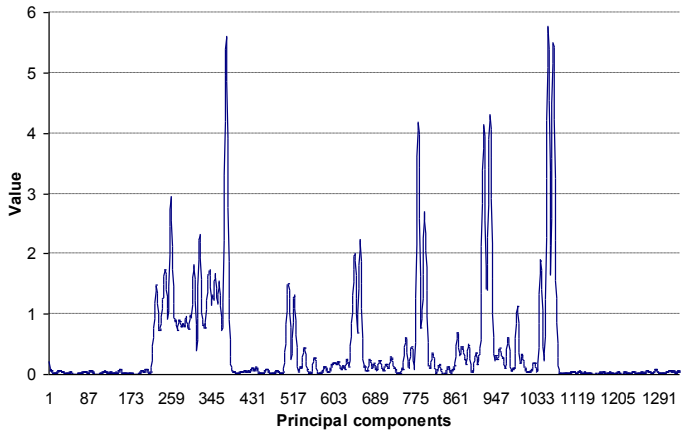


Fig. 7. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=12 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

Around the sample 400 the subject moved to the position *P1* and stopped. Although the principal components in that position have low power, the most of them exceed the detection bound. The power of principal components is lower in the *P1*, because only several links are affected with the human body.

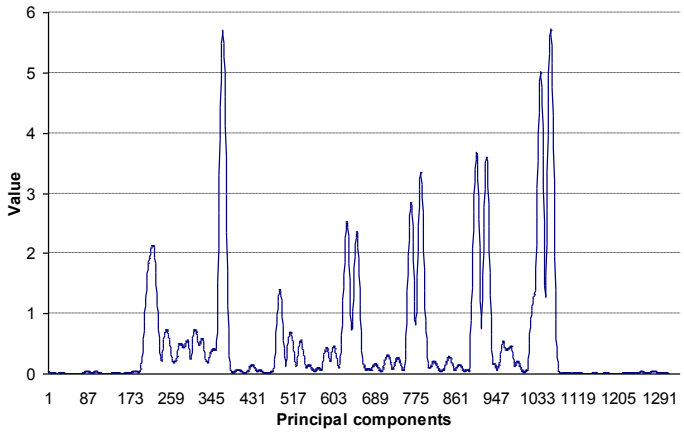


Fig. 8. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=24 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

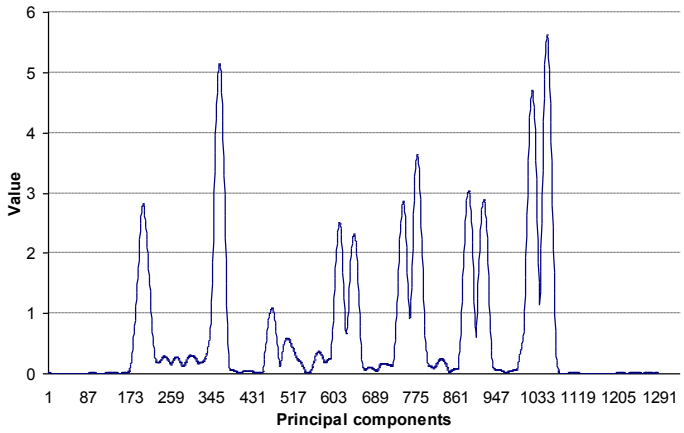


Fig. 9. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=36 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

In the Fig. 8, the shifting interval is comprised of 24 samples. The detection accuracy is approx. 93.6% for human presence. The empty room detection is 100% accurate. This shifting interval is more robust to false detections. In Fig. 9, the shifting interval is comprised of 32 samples, which is

the most robust to errors and the human presence detection is 97.8% accurate, whereas empty room is detected in 100% cases.

The false detection rate decreases as the number of samples of the interval *p* grows, but the processing time is increasing for the calculation of principal components. For *p* is 24 and 36, the polling time of 100ms for each outlet is insufficient, because the algorithm can not extract the eigenvalues in 400ms, which is the time period until the next polling cycle. From the number of experiments, the optimal polling time per outlet is determined to be 200ms if *p* is 24, and 350ms if is *p* is 36. For *p* is 12, the polling time of 100ms is satisfactory but the false detections rate is higher. Therefore, the solution for improving the processing speed is to implement an incremental algorithm which calculates eigenvalues only by using the previously calculated principal component as a predictor. The predictor is combined with the RSSI samples stored in *n*×1 vector, where *n* is the number of links. Instead of processing *p*×*p* matrix of RSSI samples, the improved algorithm calculates principal components from the vector. The fuzzy reasoning filter [35] would be useful to additionally isolate all the values below the calculated bound and the results would become more accurate. The detailed description of the filter and the incremental algorithm are not considered in this paper.

Another approach is the definition of the matrix *X* from (4), as the matrix of raw RSSI samples, instead of standard deviations. In that case, the matrix *X* is equal to the matrix *Nod* from (2) and the definition of the matrix does not require the calculation of the standard deviation per interval *p*. In the Fig. 10 the result of the applied PCA algorithm for the interval *p* which includes 12 raw samples is shown. Human presence detection accuracy is 76.4% and the detection of the empty room is accurate in 99.6% cases.

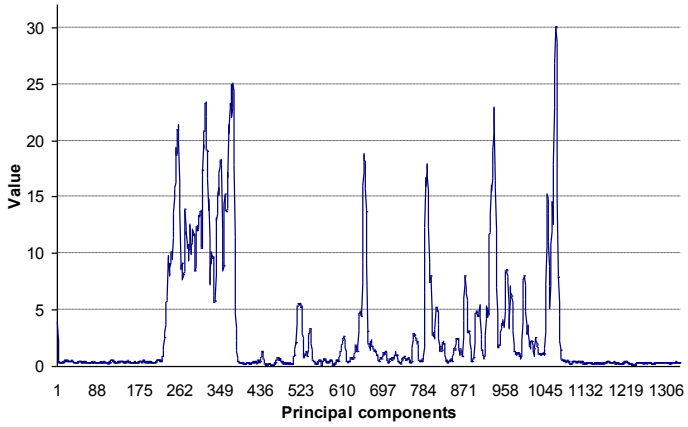


Fig. 10. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=12 and the input data representing the matrix *X* which contains raw RSSI samples

In the Fig. 11 and Fig. 12 the extracted principal components for raw RSSI inputs and *p* defined to be 24 and 36 are shown, respectively. The false

detection rate decreases as the interval *p* grows. For *p*=24, human detection accuracy is 94.2% and the empty room detection accuracy is 100%, whereas for *p*=36, human detection accuracy is 97.9%, and the empty room detection accuracy is 100%. Unfortunately, the same issue with the increased latency of the system response exists. However, the detection accuracy increases when raw values are used instead of standard deviations. The detection accuracy of these two types of inputs for the variation of the *p* interval is shown in Fig. 13.

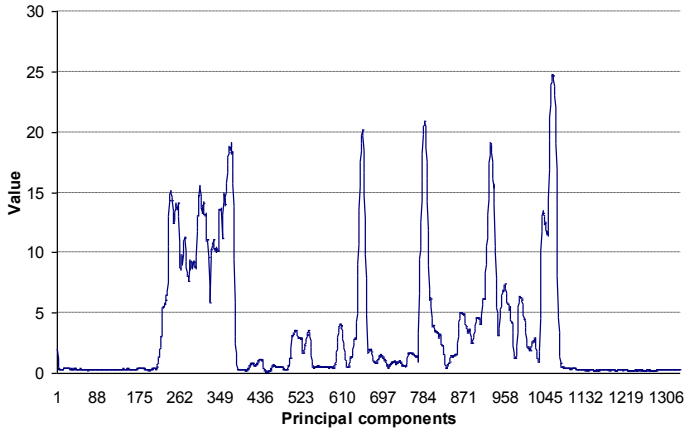


Fig. 11. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=24 and the input data representing the matrix *X* which contains raw RSSI samples

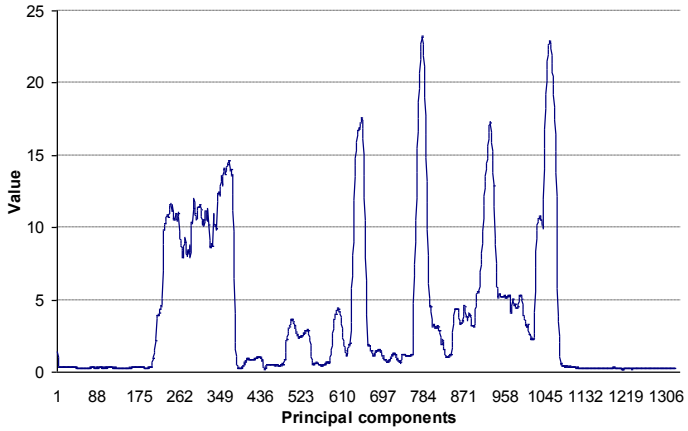


Fig. 12. PCA analysis results for R1 - an array of principal components for the shift interval defined to be *p*=36 and the input data representing the matrix *X* which contains raw RSSI samples

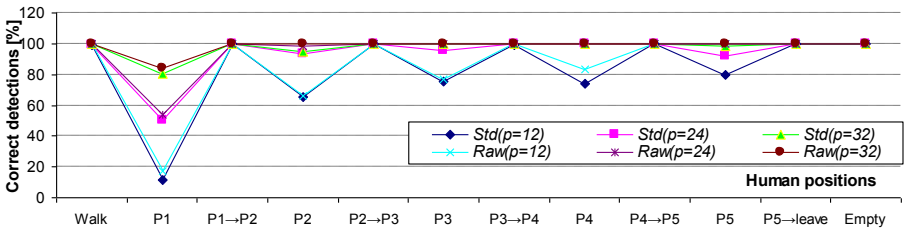
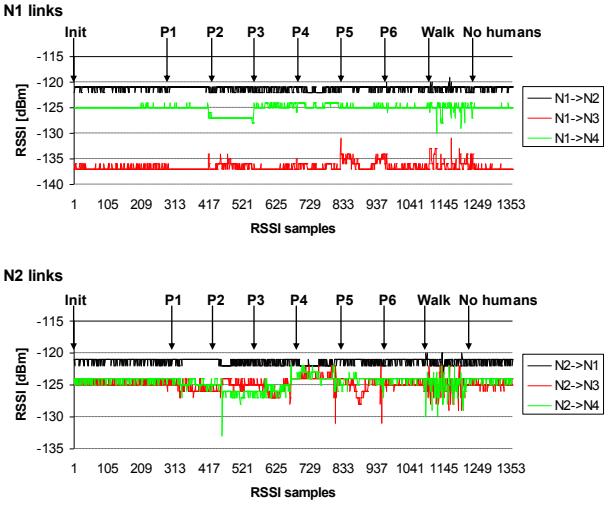


Fig. 13. The detection accuracy is shown for both standard deviations and raw values, including *p*=12, 24 and 36. The accuracy is given for human presence detection during: walking, standing in positions *P1-P5*, and transition from *Pn* to *Pn*+1 (where *n* counts from 1 to 5). The detecion of the empty room is shown as well

The test scenario in R2 was slightly different from the previous one. The room R2 was empty for a period of two minutes, and no detection was reported. Once a human stepped into the room, he was standing in each position *P1-P6* from the Fig. 5-right for one minute without movements. After samples from all positions were collected, the subject performed one minute of walking within the room by passing the positions *P1-P6*. Because of the walls' structure, this environment formed a Faraday's cage. The signal was interfered with the reflection by the walls and the RSSI variation is noticed even in the empty room. Sets of raw RSSI samples retrieved from each link between nodes, before PCA processing are logged and presented in Fig. 14.



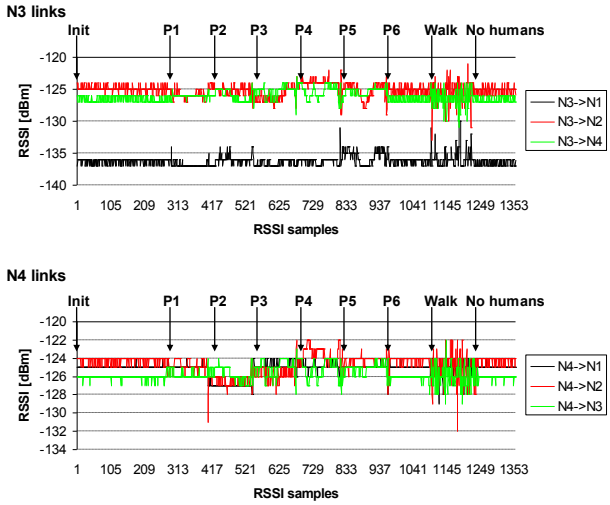


Fig. 14. Raw RSSI samples data observed in the second experimental room – R2

In the position *P1*, small RSSI variations were present on several links. In the position *P2*, the human body shadowed the links between nodes *N1* and *N4*, so the most of the signal strength was absorbed. The position *P2* had slight influence to the links *N1→N3*, *N3→N1*, *N2→N4* and *N4→N2* that were distorted by the vicinity of human body which induced the signal reflection. The human position *P3* was responsible for signal reflection between nodes *N2* and *N4* and also for the links *N2→N3* and *N3→N4* in both directions. The position *P4* caused very strong signal reflection for links between nodes *N2* and *N4*, and also *N3* and *N4*, whereas the signals between nodes *N2* and *N3* were absorbed. The human body position *P5* induced RSSI variations on links between nodes *N1* and *N3*. The strongest impact on the signal strength in the position *P5* was noticed for the links between nodes *N2* and *N3*. The influence of the position *P5* in combination with the wall reflection was responsible for the increased RSSI variation. The position *P6*, which was the furthest position from all nodes, did not affect the RSSI. Therefore, human presence detection was not possible. The position *P6* is defined as the "blind position", which is out of the detection scope. At the end of the experiment the human was walking around the room, by moving closer to nodes *N2*, *N3* and *N4*, and radio links therein, without obstructing the line-of-sight between nodes *N1* and *N2*. After one minute of walking, the room was empty, as it was at the beginning of the experiment, for a minute.

After the matrix *X* is created by using (4) and the covariance matrix C_X (7) is calculated by using standard deviations of the RSSI samples for specific interval *p*, the principal components are extracted and stored into the *Feature Vector fv*. The graphical presentations of the principal components

for 1420 samples (568 seconds) with different values of the interval *p*, are shown in Fig. 15, Fig. 16 and Fig. 17.

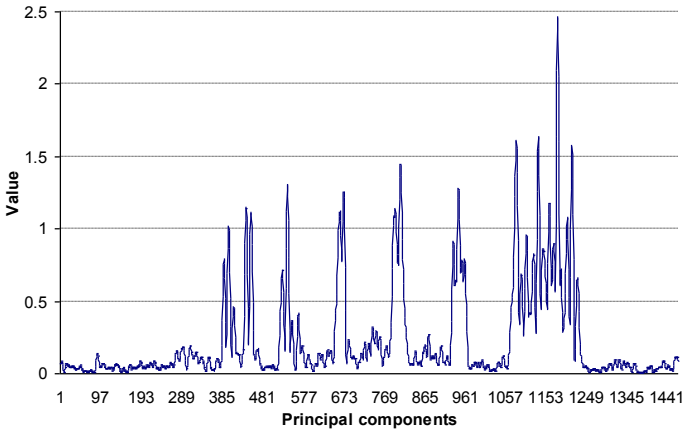


Fig. 15. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=12 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

For principal components around the samples 500 and 1000, from the Fig. 15, a higher probability of false detections occurs. The detection bound is also calculated during the initial 200 samples when the room was empty. The detection accuracy using PCA in R2 with *p* defined to include 12 samples is around 53.9% for presence, and 100% for the empty room detection.

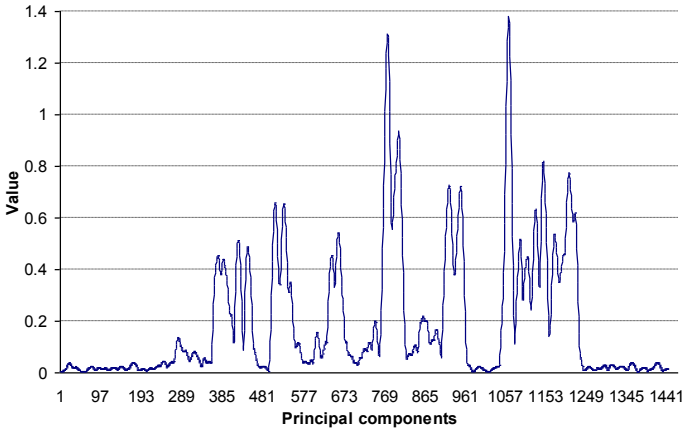


Fig. 16. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=24 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

In the Fig. 16 and Fig. 17, the shifting interval is comprised of 24 and 36 samples, respectively. For the case when *p* is 24 samples the detection

accuracy is around 81.6% for presence and 100% for the empty room. For the case when *p* is 36 samples the detection accuracy is around 90.4% for presence and 100% for the empty room.

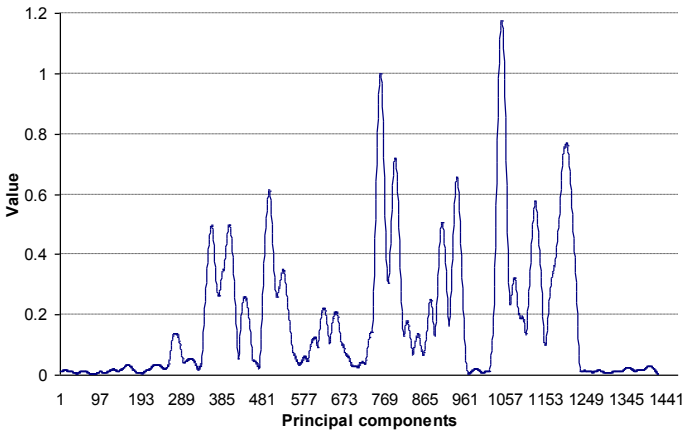


Fig. 17. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=36 and the input data representing the matrix *X* which contains standard deviations of the RSSI samples

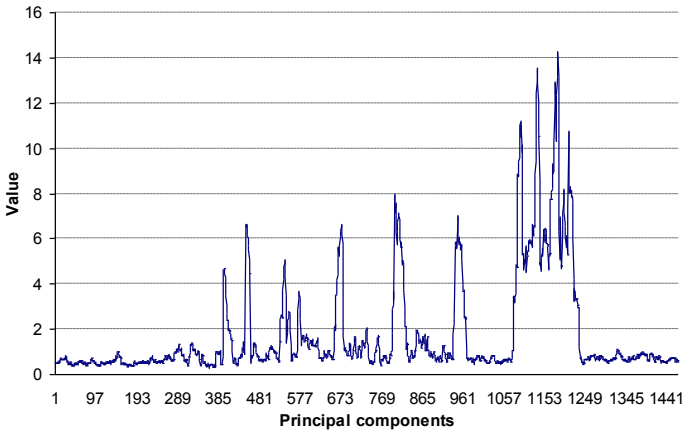


Fig. 18. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=12 and the input data representing the matrix *X* which contains raw RSSI samples

The wall reflection which was interfered with the reflection by the human body mostly affected signals in this environment. Although the initial radio map was disturbed in this environment, human presence and motion were successfully recognized for most of the positions. Only for the “blind position”

P6, the detection accuracy was very low. The integration of the additional outlets would improve the radio coverage in large rooms.

Another approach is the definition of the matrix *X* (4) as the matrix of raw samples, as in the previous experiment. In Fig. 18 the result of the applied PCA algorithm for the shifting interval *p* which includes 12 raw samples is shown. The human presence detection accuracy is 54.1% and the accuracy of the empty room detection is 99.1%. In Fig. 19 and Fig. 20, the result of the applied PCA for the interval *p* which includes 24 and 36 raw samples is shown, respectively. The false detection rate decreases as the number of *p* samples grows, but larger *p* implicates longer latency which should be optimized with an iterative method combined with the fuzzy filter.

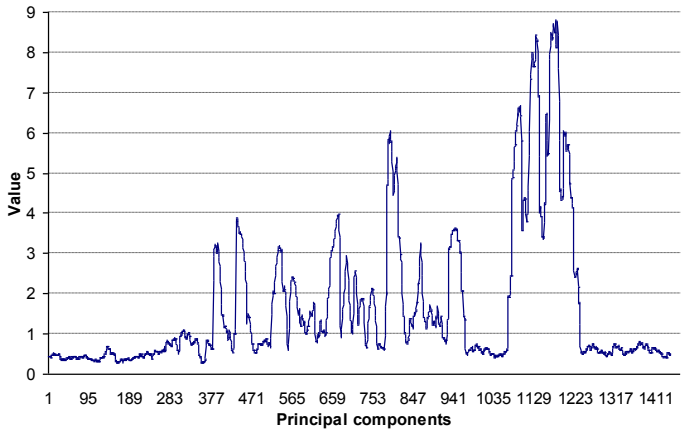


Fig. 19. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=24 and the input data representing the matrix *X* which contains raw RSSI samples

The human presence detection accuracy for the *p* interval of 24 samples is approx. 83%, whereas the accuracy for the empty room detection is 100%. The human presence detection accuracy for *p*=36 samples is 91.5%, whereas the empty room detection accuracy is 100%. The detailed error distribution, depending of *p* and the input samples, is shown in Fig. 21.

As concluded for the previous experiment, the same conclusion can be deduced for this experiment: the detection accuracy increases when using raw RSSI values, in contrary to standard deviations of the RSSI.

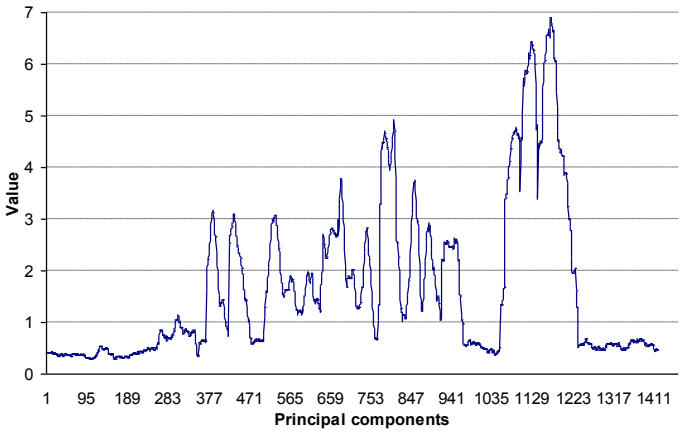


Fig. 20. PCA analysis results for R2 - an array of principal components for the shift interval defined to be *p*=36 and the input data representing the matrix *X* which contains raw RSSI samples

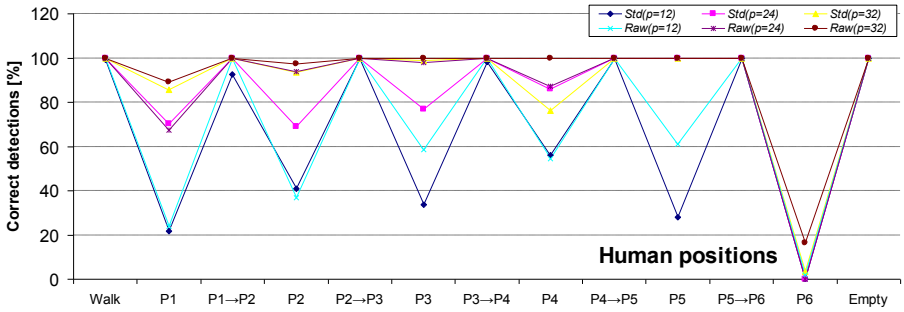


Fig. 21. The detection accuracy is shown for both standard deviations and raw values, including *p*=12, 24 and 36. The accuracy is given for human presence detection during: walking, standing in positions *P1-P6*, and moving from *Pn* to *Pn*+1 (where *n* is 1 to 6). The detecion of the empty room is also shown

5.1. Energy Saving Experiment

The integration of the provided energy-saving system [5] with the proposed method for human presence detection, enabled a prototype realization. The prototype has been installed in four, the least frequently occupied rooms in an average household. The primary goal was to demonstrate the proposed method operating in real conditions for the energy saving. Energy saving has been achieved by utilizing two approaches: (1) if there is nobody present in the room for more than 10 seconds, turn the light in that room off, (2) always

decrease the brightness of lights in the household by 10%, what should be unnoticeable to users but saves an amount of energy. To be able to provide the comparison between the regular human behavior on energy saving and the proposed prototype, 8 bulbs of 100W combined with 12 smart outlets and 4 smart light switches have been installed in each room (4 smart nodes and 2 bulbs per a room). In each room, one bulb was under regular control (manual on/off switching), which included the worst case - a user leaves the light on after leaving a room. The second bulb was under automatic control. The automatic control is achieved by using predefined power behavior schemes, which define system responses on human presence detection events. Operational "energy saving" mode switched off the light after 10 seconds when no-presence was reported by the RSSI analyzer module (from Fig. 2), and also switched the light on, almost immediately, when a human entered the room. In each room, both bulbs (lamps) were plugged to smart outlets in order to provide the power consumption logging for the detailed comparison.

The experiment has been performed during one working day with four-member family (two adults and two kids). Two bedrooms, one bathroom and a foyer have been defined as test rooms, where the real presence of humans was the most dynamic. The test subjects performed the normal behavior at home, trying to manually switch off the lights in each unoccupied room. All the rooms were properly covered with the radio signal and no "blind positions" were recognized. The walls were made of concrete and brick blocks, 30cm thick for exterior walls and 20cm for interior wall.

Supported with the proposed presence detection algorithm, the energy consumption used for lights was decreased from 1220 W/h to 730 W/h at the end of the day. In the Fig. 22, the power consumption, achieved by using regular and automatic control is shown, per each hour during the experiment.

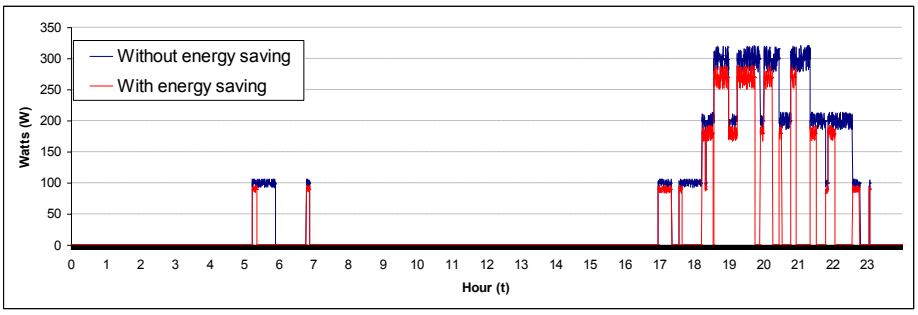


Fig. 22. Measured power consumption in the experimental house by comparing manual interaction for lights control (blue) and automatic "energy saving mode" (red)

For this experiment, PCA used inputs of p=24 raw samples. With the polling time of 200ms per an outlet, the detection could be reported on each 800ms, including additional 600ms for the PCA processing and the generation of a functional status. This is the optimal time with the high accuracy, suitable for this experiment. For p=36 samples, the functional

status can be generated after approx. 2.8s (including the polling cycle for detection), which is too long. The polling time of 200ms is acceptable when the number of smart nodes is four or less. For additional smart nodes, the polling time has to be reduced to 100ms. The detection accuracy for p=12 (100ms polling) is not optimal.

6. Conclusion

PCA presents a simple method for extracting relevant information from confusing and large data sets. It is a variable reduction procedure and is useful when samples are obtained on a large number of variables that are mutually correlated. PCA helps identifying patterns in the data, and expressing the data in a way that highlights their similarities or differences. Because of this variables redundancy, it is possible to reduce a large set of observed variables into a smaller number of principal components, while retaining as much as possible of the variation present in the original data set.

The presented article confirms the hypothesis that human presence detection is possible by applying the PCA to the set of RSSI samples obtained from radio links between wireless power outlets. The experimental results show that PCA inputs, given in a form of raw RSSI samples, provide more accurate results for human presence detection, than the inputs which describe the dispersion of the signal, such as standard deviation. More accurate detection requires larger set of input samples, which implicates larger processing time and overall system response delay. For the future improvement, the testing would be performed in another buildings made of different materials, with dynamic changes of the furniture layout which can introduce additional interferences (noise) to the system. The testing in such environments would be helpful for finding patterns that would enable the definition of a fuzzy reasoning algorithm which would improve the accuracy of human presence detection. Aditionally, an incremental method needs to be defined which would speed up the processing time, and the overall system's response for the larger number of additional wireless smart nodes.

The presented solution can significantly conserve electric energy in a household, by executing automatic operations which switch off power on devices that are not used for a specified period of time. The test subjects confirmed that 1s to 1.5s are acceptable for the functional status generation. However, the further intention is to decrease the system's response without decreasing the accuracy of the proposed algorithm.

Acknowledgement. This work was partially supported by the Ministry of Education and Science of the Republic of Serbia under Grant TR-32029 and by the Secretary of Science and Technology Development of the Province of Vojvodina under Grant 114-451-2434/2011-03.

References

1. Jahn, M., Jentsch, M., Prause, C.R., Pramudianto, F., Al-Akkad, A. and Reiners, R.: The energy aware smart home. In Proceeding of the 5th International Conference on Future Inform. Technologies (FutureTech '10), Korea, 1-8. (2010)
2. Sundramoorthy, V., Liu, Q., Cooper, G., Linge, N. and Cooper, J.: DEHEMS: a user-driven domestic energy monitoring system. In Proceedings of Internet of Things (IOT '10), 1-8. (2010)
3. Song, G., Ding, F., Zhang, W. and Song, A.: A wireless power outlet system for smart homes. IEEE Transactions on Consumer Electronics, Vol. 54, No. 4, 1688-1691. (2008)
4. Han, J., Lee, H. and Park, K.R.: Remote-controllable and energy-saving room architecture based on ZigBee communication. IEEE Transactions on Consumer Electronics, Vol. 55, No. 1, 264-268. (2009)
5. Mrazovac, B., Bjelica, M.Z., Papp, I. and Teslic, N.: Towards ubiquitous smart outlets for safety and energetic efficiency of home electric appliances. In Proceedings of International Conference on Consumer Electronics (ICCE '11), Berlin, Germany, 324-328. (2011)
6. European Regulators' Group for Electricity and Gas, Smart Metering with a Focus on Electricity Regulation, Document E07- RMF-04-03. (2007)
7. Ababneh, N.: Radio irregularity problem in wireless sensor networks: New experimental results. In Proceedings of IEEE Sarnoff Symposium, Princeton, USA, 1-5. (2009)
8. Zhou, G., He, T., Krishnamurthy, S., and Stankovic, J.A.: Impact of radio irregularity on wireless sensor networks. In Proceedings of the 2nd International Conference on Mobile Systems, Applications and Services (MobiSys '04), 125-138. (2004)
9. Youssef, M., Mah, M., and Agrawala, A.: Challenges: device-free passive localization for wireless environments. In Proceedings of the 13th annual ACM International Conf. on Mobile Computing and Networking, pp. 222–229 (2007)
10. Woyach, K., Puccinelli, D. and Haenggi, M.: Sensorless sensing in wireless networks: Implementation and measurements. In Proc. of 2nd Intl. Workshop on Wireless Network Measurement (WinMee'06), Boston, USA, 1-8. (2006)
11. Puccinelli, D., Foerster, A., Puiatti, A. and Giordano, S.: Radio-Based Trail Usage Monitoring with Low-End Motes. In Proceedings of the 7th IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing (PerSens '11), Seattle, USA, 196-201. (2011)
12. Lee, P.W.Q., Seah, W.K.G., Tan, H.P. and Yao, Z.X.: Wireless Sensing without Sensors - An experimental study of motion/intrusion detection using RF irregularity. Journal of Measurement Science and Technology, Special Issue on Wireless Sensor Networks: Designing for real-world deployment and deployment experiences, Vol. 21, No. 12, (2010)
13. Lin, W.C., Seah, W.K.G. and Li, W.: Exploiting radio irregularity in the internet of things for automated people counting. In Proc. of the 22nd IEEE Symposium on Personal, Indoor, Mobile and Radio Comm. (PIMRC '11), Toronto Canada, (2011)
14. Hussain, S., Peters, R. and Silver, D.L.: Using received signal strength variation for surveillance in residential areas. In Proc. of the 9th ACM/IEEE Intl. Conf. on Information Processing in Sensor Networks (IPSN '10), Vol. 6973, 1-6. (2008)
15. Kaltiokallio, O., Bocca, M. and Eriksson, L.: Distributed RSSI processing for intrusion detection in indoor environments. In Proc. of 9th ACM/IEEE

International Conf. on Information Processing in Sensor Networks (IPSN ’10), 404-405. (2010)
16. Puzo, E.L., Brenner, R.P., Walker, T.O. and Anderson, C.R.: The Matrix: a roadside wireless security system. In Proceedings of Virginia Tech Symposium on Wireless Personal Communications, Virginia, USA, (2011)
17. Patwari, N. and Wilson, J.: RF sensor networks for device-free localization and tracking. In Proceedings of the IEEE, Vol. 98, No. 11, 1961-1973. (2010)
18. Deak, G., Curran, K. and Condell, J.: History Aware Device-free Passive (DfP) Localisation. Image Processing & Communications Journal, vol. 16, no. 3-4, pp. 21-30. (2011)
19. Zhang, D., Ma, J., Chen, Q. and Ni, L. M.: An RF-Based System for Tracking Transceiver-Free Objects. In Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications, (PerCom ‘07), pp. 135-144. (2007)
20. Zhang, D. and Ni, L. M.: Dynamic Clustering for Tracking Multiple Transceiver-Free Objects. In Proceedings of the Seventh Annual IEEE International Conf. on Pervasive Computing and Communications (PerCom ‘09), pp. 1-8. (2009)
21. Moussa, M. and Youssef, M.: Smart Cevices for Smart Environments: Device-free Passive Detection in Real Environments. In Proceedings of the Seventh Annual IEEE International Conference on Pervasive Computing and Communications (PerCom ‘09), pp.1-6. (2009)
22. Shah, N., Tsai, C.-F. and Chao, K.-M.: Monitoring appliances sensor data in home environment: Issues and challenges. In Proceedings of IEEE Conference on Commerce and Enterprise Computing (CEC '09), 439-444. (2009)
23. Chen, C.-S. and Lee, D.-S.: Energy saving effects of wireless sensor networks: A case study of convenience stores in Taiwan, Sensors '11, Vol.11, No.2, 2013-2034. (2011)
24. Chao, K.-M., Shah, N., Matei, A., Zlamaniec, T., Li, W, Lo, C.-C. and Li, Y.: Intelligent interactive system for collaborative green computing. In Proceedings of CSCWD ‘11, 690-697. (2011)
25. Pensas, H., Raula, H. and Vanhala, J.: Energy efficient sensor network with service discovery for smart home environments. In Proc. of SENSORCOMM '09, 399-404. (2009)
26. Zhou, Z., Xiang, X. and Wang, X.: An energy-efficient data-dissemination protocol in wireless sensor networks. In Proc. of International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM’06), 13-22. (2006)
27. Mrazovac, B., Bjelica, M.Z., Todorovic, B.M., Miljkovic, A. and Samardzija, D.: Using Radio Irregularity for Increasing Residential Energy Awareness. Telfor Journal, vol. 4, no. 1, pp. 31-36. (2012)
28. Mrazovac, B., Bjelica, M.Z., Kukolj, D., Todorovic, B.M. and Samardzija, D.: A Human Detection Method for Residential Smart Energy Systems Based on Zigbee RSSI Changes. IEEE Transactions on Consumer Electronics, vol.58, no.3, pp. 819-824. (2012)
29. Smith, L.I.: A tutorial on Principal Components Analysis. (2002). [Online]. Available: http://www.sccg.sk/~haladova/principal_components.pdf (May 2012)
30. Shlens, J.: A tutorial on Principal Component Analysis: Derivation, Discussion and Singular Value Decomposition. (2003). Available: http://www.cs.princeton.edu/ picasso/mats/PCA-Tutorial-Intuition_jp.pdf (May 2012)

31. Castells, F., Lasaosa, P.L., Sörnmo, L., Bollmann, A. and Millet-Roig, J.: Principal Component Analysis in ECG Signal Processing. EURASIP Journal on Advances in Signal Processing. Vol. 2007, Article ID 74580. (2007)
32. Jain, A., Duin, R. and Mao, J.: Statistical Pattern Recognition: A Review. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. 22, No. 1, 4-37. (2000)
33. Bjelica, M.Z., Mrazovac, B. and Teslic, N.: Evaluation of the Available Scripting Languages for Home Automation Networks: Real World Case Study. In Proc. of TELSIKS'11, Nis, Serbia, 611-614. (2011)
34. Mrazovac, B., Bjelica, M.Z., Teslic, N., Papp, I., Temerinac, M.: Consumer-oriented Smart Grid for Energy Efficiency. In Proc. of VDE Kongress 2012, Stutgart, Germany (2012)
35. Sánchez-Torrubia, M.G. and Torres-Blanc C.: GRAPHs: A Mamdani-type Fuzzy Inference System to Automatically Assess Dijkstra's Algorithm Simulation. International Journal on Information Theory and Applications, Vol. 17, No. 1, 35-48. (2010)

Bojan Mrazovac has received BSc and MSc degrees from the Faculty of Technical Sciences, University of Novi Sad, Serbia. He is currently employed as a software design engineer at RT-RK LLC, and he is also working towards his PhD degree at the department for Computer Engineering and Computer Communications, at Faculty of Technical Sciences, Novi Sad. He is currently enrolled in several research projects including home automation and energy aware systems. His expertise also covers the range of embedded platforms and software with the emphasis on video and radio signal processing.

Milan Z. Bjelica has received BSc and MSc degrees from the Faculty of Technical Sciences, University of Novi Sad, Serbia. He is currently employed as a software design engineer at RT-RK LLC, and as the teaching assistant at the University of Novi Sad. He is also working towards his PhD degree at the department for Computer Engineering and Computer Communications, at Faculty of Technical Sciences, Novi Sad. His research interests and expertise include novel immersive context-aware systems, usability assessment and home automation systems, frequently applied to the different consumer electronics devices.

Dragan Kukolj has received his Diploma degree in control engineering in 1982, MSc degree in computer engineering in 1988, and PhD degree in control engineering in 1993, all from the University of Novi Sad, Serbia. He is currently a Professor of computer-based systems with Department of Computing and Control, Faculty of Engineering, University of Novi Sad. His main research interests include soft computing, data mining techniques and computer-based systems integration with applications in video processing and digital signal processing. He has published over 100 papers in referred journals and conference proceedings. He is the coordinator of Intellectual Property Centre of Faculty of Engineering.

Branislav M. Todorović received his Dipl.Eng. and M.Sc. degrees from the Faculty of Electrical Engineering, University of Belgrade, Serbia and Ph.D. degree from the Faculty of Technical Sciences, University of Novi Sad, Serbia in 1983, 1988 and 1997, respectively. He is a Senior research fellow at the RT-RK, Institute for Computer Based Systems, and a Professor at the Military Academy, University of Defense, Belgrade. Prior to joining RT-RK, he was with the Institute of Microwave Techniques and Electronics (IMTEL), Centre for Multidisciplinary Research and the Military Technical Institute (VTI, Institute of Electrical Engineering) in Belgrade. His research interests are in the wide area of radio signals, telecommunications and digital signal processing. He has authored or coauthored about 100 referred journal and conference papers and two books. He is a Member of the IEEE and Fellow of the Institute of Nanotechnology (UK).

Saša Vukosavljev has received his BSc and MSc degree from the Faculty of Technical Sciences, University of Novi Sad, Serbia. He is currently employed as a senior embedded software developer in RT-RK, Institute for Computer Based System, Novi Sad. His main research interests include low power embedded real time systems, lightweight sensors, signal processing and autonomous robots. He took part in European competition in robotics in France in 2001 and 2002. He has authored and coauthored about 30 conference papers.

Received: May 31, 2012; Accepted: November 12, 2012.

DOI: 10.2298/CSIS120614011M

Support for End-to-End Response-Time and Delay Analysis in the Industrial Tool Suite: Issues, Experiences and a Case Study*

Saad Mubeen[1], Jukka Mäki-Turja[1,2], and Mikael Sjödin[1]

[1] Mälardalen Real-Time Research Centre (MRTC), Mälardalen University, Sweden
[2] Arcticus Systems, Järfälla, Sweden
{saad.mubeen, jukka.maki-turja, mikael.sjodin}@mdh.se

Abstract. In this paper we discuss the implementation of the state-of-the-art end-to-end response-time and delay analysis as two individual plug-ins for the existing industrial tool suite Rubus-ICE. The tool suite is used for the development of software for vehicular embedded systems by several international companies. We describe and solve the problems encountered and highlight the experiences gained during the process of implementation, integration and evaluation of the analysis plug-ins. Finally, we provide a proof of concept by modeling the automotive-application case study with the existing industrial model (the Rubus Component Model), and analyzing it with the implemented analysis plug-ins.

Keywords: real-time systems, response-time analysis, end-to-end timing analysis, component-based development, distributed embedded systems.

1. Introduction

Often, an embedded system needs to interact and communicate with its environment in a timely manner, i.e., the embedded system is a real-time system. For such a system, the desired and correct output is one which is logically correct as well as delivered within a specified time. The safety-critical nature of many real-time systems requires evidence that the actions by them will be provided in a timely manner, i.e., each action will be taken at a time that is appropriate to the environment of the system. Therefore, it is important to make accurate predictions of the timing behavior of these systems.

In order to provide evidence that each action in the system will meet its deadline, *a priori* analysis techniques such as schedulability analysis have been developed by the research community. Response Time Analysis (RTA) [17, 45] is one of the methods to check the schedulability of a system. It calculates upper bounds on the response times of tasks or messages in a real-time system or a network respectively. Holistic Response-Time Analysis (HRTA) [48, 47, 42]

* This work is supported by the Swedish Knowledge Foundation (KKS) within the projects FEMMVA and EEMDEF. The authors thank the industrial partners Arcticus Systems, BAE Systems Hägglunds and Volvo Construction Equipment Sweden.

is an academic well established schedulability analysis technique to calculate upper bounds on the response times of task chains that may be distributed over several nodes in a Distributed Real-time Embedded (DRE) system.

A task chain is a sequence of more than one task in which every task (except first) receives a trigger, data or both from its predecessor. One way to classify these chains is as trigger and data. In trigger chains, there is only one triggering source (e.g, event, clock or interrupt) that activates the first task. The rest of the tasks are activated by their predecessors. In data chains, tasks are activated independent of each other, often with distinct periods. Each task (except the first) in these chains receives data from its predecessor. The first task in a data chain may receive data from the peripheral devices and interfaces, e.g., signals from the sensors or messages from the network interfaces. The end-to-end timing requirements on trigger chains are different from those on data chains. If a system is modeled with trigger chains only, it is called a single-rate system. On the other hand, if the system contains at least one data chain with different clocks then the system is said to be multi-rate.

The end-to-end delays should also be computed along with the holistic response times to predict complete timing behavior of multi-rate real-time systems [21]. For this purpose, the research community has developed the End-to-End[3] Delay Analysis (E2EDA). In [21], the authors have a view that almost all automotive embedded systems are multi-rate systems. The industrial tools used for the development of these systems should be equipped with the state-of-the-art timing analysis.

A tool chain for the industrial development of component-based DRE systems consists of a number of tools such as designer, compiler, builder, debugger, simulator, etc. Often, a tool chain may comprise of tools that are developed by different tool vendors. The implementation of state-of-the-art complex real-time analysis techniques such as RTA, HRTA and E2EDA in such a tool chain is non-trivial because there are several challenges that are encountered apart from merely coding and testing the analysis algorithms. These challenges and corresponding solutions that we propose are central to this paper.

Goals and Contributions. In this paper[4], we discuss the implementation of HRTA and E2EDA as two individual plug-ins in the existing industrial tool suite Rubus-ICE (Integrated Component development Environment) [1]. Our goal is to transfer the state-of-the-art real-time analysis results, i.e., HRTA and E2EDA to the existing tools for the industrial use. We discuss and solve the problems encountered, solutions proposed and experiences gained during the implementation, integration and evaluation of the plug-ins. We also provide a proof of concept by conducting the automotive-application case study. These new plug-ins support complete end-to-end timing analysis of DRE systems. Thus, the scope and usability of Rubus tools has widened with the addition of these plug-ins.

[3] The terms “holistic” and “end-to-end” mean the same thing. In order to be consistent with the previous work and naming conventions used in the existing industrial tools, we will use “holistic” with response-times and “end-to-end” with delays.

[4] This work is the extension of our previous work [37].

Paper Layout. Section 2 presents the background and related work. Section 3 discusses the end-to-end timing requirements and the implemented analysis. Section 4 describes the challenges encountered, solutions proposed and experiences gained during the implementation and integration of the plug-ins. In Section 5, we present a case study by modeling and analyzing the automotive DRE application. Section 6 concludes the paper and presents the future work.

2. Background and Related Work

2.1. The Rubus Concept

Rubus is a collection of methods and tools for model- and component-based development of dependable embedded real-time systems. Rubus is developed by Arcticus Systems [1] in close collaboration with several academic and industrial partners. Rubus is today mainly used for development of control functionality in vehicles by several international companies [2, 13, 7, 5]. The Rubus concept is based around the Rubus Component Model (RCM) [27] and its development environment Rubus-ICE, which includes modeling tools, code generators, analysis tools and run-time infrastructure. The overall goal of Rubus is to be aggressively resource efficient and to provide means for developing predictable and analyzable control functions in resource-constrained embedded systems.

RCM expresses the infrastructure for software functions, i.e., the interaction between them in terms of data and control flow separately. The control flow is expressed by triggering objects such as internal periodic clocks, interrupts and events. In RCM, the basic component is called Software Circuit (SWC). Its execution semantics are: upon triggering, read data on data *in-ports*; execute the function; write data on data *out-ports*; and activate the output trigger.

RCM separates the control flow from the data flow among SWCs within a node. Thus, explicit synchronization and data access are visible at the modeling level. One important principle in RCM is to separate functional code and infrastructure implementing the execution model. RCM facilitates analysis and reuse of components in different contexts (SWC has no knowledge how it connects to other components). The component model has the possibility to encapsulate SWCs into software assemblies enabling the designer to construct the system at different hierarchical levels. Recently, we extended RCM for the development of DRE systems by introducing new components [30, 39, 33]. A detailed comparison of RCM with several component models is presented in [39].

Fig. 1(a) depicts the sequence of main steps followed in Rubus-ICE from modeling of an application to the generation of code. An application is modeled in the Rubus Designer tool. Then the compiler compiles the design model into the Intermediate Compiled Component Model (ICCM). After that the builder tool sequentially runs a set of plug-ins. Finally, the coder tool generates the code.

2.2. The Rubus Analysis Framework

The Rubus model allows expressing real-time requirements and properties at the architectural level. For example, it is possible to declare real-time require-

ments from a generated event and an arbitrary output trigger along the trigger chain. For this purpose, the designer has to express real-time properties of SWCs, such as worst-case execution times and stack usage. The scheduler will take these real-time constraints into consideration when producing a schedule. For event-triggered tasks, response-time calculations are performed and compared to the requirements. The analysis supported by the model includes response time analysis and shared stack analysis.

2.3. Plug-in Framework in Rubus-ICE

The plug-in framework in Rubus-ICE [26] facilitates the implementation of research results in isolation (without needing Rubus tools) and their integration as add-on plug-ins (binaries or source code) with the Rubus-ICE. A plug-in is interfaced with the builder tool as shown in Fig. 1(a). The plug-ins are executed sequentially which means that the next plug-in can execute only when the previous plug-in has run to completion. Hence, each plug-in reads required attributes as inputs, runs to completion and finally writes the results to the ICCM file. The Application Programming Interface (API) defines the services required and provided by a plug-in. Each plug-in specifies the supported system model, required inputs, provided outputs, error handling mechanisms and a user interface. Fig. 1(b) shows the conceptual organization of a plug-in in the Rubus-ICE.

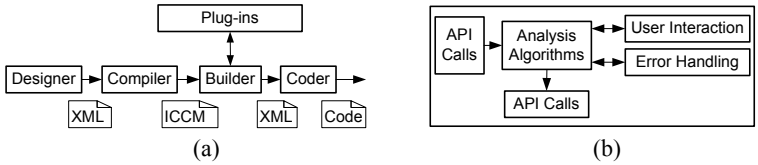


Fig. 1. Sequence of steps from design to code generation in Rubus-ICE

2.4. Response-Time Analysis

RTA of Tasks in a Node. Liu and Layland [28] provided theoretical foundation for analysis of fixed-priority scheduled systems. Joseph and Pandya published the first RTA [25] for the simple task model presented in [28]. Subsequently, it has been applied and extended in a number of ways by the research community. RTA applies to systems where tasks are scheduled with respect to their priorities and which is the predominant scheduling technique used in real-time operating systems [40]. Tindell [47] developed the schedulability analysis for tasks with offsets for fixed-priority systems. It was extended by Palencia and Gonzalez Harbour [42]. Later, Mäki-Turja and Nolin [29] reduced pessimism from RTA developed in [47, 42] and presented a tighter RTA for tasks with offsets by accurately modeling inter-task interference. We implemented tighter version of RTA of tasks with offsets [29] as part of the HRTA and E2EDA.

RTA of Messages in a Network. To stay focussed on the automotive or vehicular domain, we will consider only Controller Area Network (CAN) and its high-level protocols. Tindell et al. [49] developed the schedulability analysis of

CAN which has served as a basis for many research projects. Later on, this analysis was revisited and revised by Davis et al. [19]. The analysis in [49, 19] assumes that all CAN device drivers implement priority-based queues. In [20] Davis et al. pointed out that this assumption may become invalid when some nodes in a CAN network implement FIFO queues. Hence, they extended the analysis of CAN with FIFO queues as well. In this work, the message deadlines are assumed to be smaller than or equal to the corresponding periods. In [18], Davis et al. lifted this assumption by supporting the analysis of CAN messages with arbitrary deadlines. Furthermore, they extended their work to support RTA of CAN for FIFO and work-conserving queues.

However, the existing analysis does not support mixed messages which are implemented by several high-level protocols for CAN. In [32, 36, 31], Mubeen et al. extended the existing analysis to support RTA of mixed messages in the CAN network where some nodes use FIFO queues while others use priority queues. Later on, Mubeen et al. [38] extended the existing analysis for CAN to support mixed messages that are scheduled with offsets in the controllers that implement priority-ordered queues. In this work we will consider all of the above analyses as part of the end-to-end response-time and delay analysis.

Holistic RTA. The holistic response-time analysis calculates the response times of event chains that are distributed over several nodes (also called distributed transactions) in a DRE system. It combines the analysis of nodes (uniprocessors) and networks. In this paper, we consider the end-to-end timing model that corresponds to the holistic schedulability analysis for DRE systems [48]. In [34], we discussed our preliminary findings about implementation issues that are encountered when HRTA is transferred to the industrial tools.

End-to-end Delay Analysis. Stappert et al. [46] formally described end-to-end timing constrains for multi-rate systems in the automotive domain. In [21], Feiertag et al. presented a framework (developed in TIMMO project [16]) for the computation of end-to-end delays for multi-rate automotive embedded systems. Furthermore, they emphasized on the importance of two end-to-end latency semantics, i.e., “maximum age of data” and “first reaction” in control systems and body electronics domains respectively. A scalable technique, based on model checking, for the computation of end-to-end latencies is described in [43]. In this work, we will implement the end-to-end delay analysis of [21].

2.5. Tools for End-to-end Timing Analysis of DRE Systems

The MAST tool suite [6] implements a number of state-of-the-art analysis algorithms for DRE systems. Among them is the offset-based analysis algorithm [47, 42] whose tighter version [29] is implemented as part of the HRTA and E2EDA plug-ins. It also allows visual modeling and analysis of real-time systems in a Unified Modeling Language (UML) design environment. The Volcano Family [10] is a bunch of tools for designing, analyzing, testing and validating automotive embedded software systems. Volcano Network Architect (VNA) [12] is a communication design tool that supports the analysis of Local Interconnect

Network (LIN) and CAN. It also supports end-to-end timing analysis of a system with more than one network. It implements RTA of CAN presented in [49].

SymTA/S [23] is a tool for model-based timing analysis and optimization. It implements several real-time analysis techniques for single-node, multiprocessor and distributed systems. It supports RTA of software functions, RTA of bus messages and end-to-end timing analysis of both single-rate and multi-rate systems. It is also integrated with the UML development environment to provide a timing analysis support for the applications modeled with UML [22].

Vector [11] is a tools provider for the development of networked electronic systems in the automotive and related domains. In the Vector tool family, CANoe [3] is a tool for the development, testing and analysis of ECU (Electronic Control Units) networks and individual ECUs. It supports various protocols for network communication including CAN, LIN, MOST, Flexray, Ethernet and J1708. Network Designer CAN is another tool by Vector that is used to design the architecture and perform timing analysis of CAN network. RAPID RMA [8] implements several scheduling schemes and supports end-to-end analysis for single- and multiple-node real-time systems. It also allows real-time analysis support for the systems modeled with Real-Time CORBA [44].

The Rubus-ICE tool suite allows a developer to specify timing information and perform the HRTA and E2EDA at the modeling phase during component-based development of DRE systems. To the best of our knowledge, Rubus-ICE is the first and only tool suite that implements RTA of mixed messages in CAN [32], RTA of mixed messages with offsets [38] and a tighter version of offset-based RTA algorithm [29] as part of the HRTA and E2EDA .

3. End-to-end Timing Requirements and Implemented Analysis in Rubus-ICE

3.1. End-to-end timing requirements in trigger chains

A real-time system (single-node or distributed) can be modeled with trigger chains (see Fig.2(a)), data chains (see Fig.2(b)) or a combination of both. The end-to-end timing requirements on trigger chains are different from those on data chains. If the system is modeled with trigger chains then the end-to-end deadline requirements are placed on the holistic response times.

An example of a trigger chain that consists of three components is shown in Fig. 2(a). Assume that each component corresponds to a task at run-time. When task τ_{SWC_A} finishes its execution, it triggers τ_{SWC_B}. Similarly, τ_{SWC_C} can only be triggered by τ_{SWC_B} after finishing its execution. There cannot be multiple outputs corresponding to a single input signal. In fact, there will always be one output of the chain corresponding to the input trigger. Hence, the end-to-end timing requirements correspond to the holistic response times. Distributed real-time systems can also be modeled with trigger chains in a similar fashion.

3.2. End-to-end timing requirements in data chains

As compared to the systems which are modeled with trigger chains, merely computing the holistic response times and comparing them with the end-to-end deadlines is not sufficient to predict the complete timing behavior of multi-rate real-time systems which are modeled with data chains. There may be over- and under-sampling in such systems because the individual tasks are activated by independent clocks, often with different periods. Since data is transferred among tasks and messages within a data chain by means of asynchronous buffers, there exist different semantics of end-to-end delay in a data chain. These buffers are often of a non-consuming type which means the data stays in the buffer after it is read by the reader task. Moreover, the data in the buffer can be overwritten by the writer task with new values before the previous value was read by the reader task. Therefore, some input values in the data buffers can be overwritten by new values, and hence the effect of the old input values may never propagate to the output of a data chain. Further, there may be several duplicates of the output of a data chain corresponding to a particular input.

The end-to-end timing requirements in multi-rate real-time systems, especially in the automotive domain, are placed on the first reaction to the input and age of the data at the output [21]. The end-to-end delay in a data chain refers to the time elapsed between the arrival of a signal at the first task and production of corresponding output signal by the last task in the chain (provided the information corresponding to the input has traversed the chain from first to last task) [43]. In a single-rate real-time system that contains only trigger chains, tasks in a chain are not activated by independent events, in fact, there is only one activating event in the chain. Hence, the holistic response times and end-to-end delays will have equal values. On the other hand, these values are not the same in multi-rate real-time systems that are modeled with data chains. Therefore, a complete analysis of a real-time system modeled with data chains requires the calculation of not only holistic response times but also end-to-end delays.

Examples. A multi-rate real-time system modeled with three SWCs in RCM is shown in Fig. 2(b). These SWCs are activated by independent clocks with different periods, i.e., 8ms, 16ms and 4ms respectively. SWC_A reads the input signals from the sensors while SWC_C produces the output signals for the actuators. Assume that each SWC will be allocated to an individual task by the run-time environment generator. Also assume that WCET of each task is $1ms$. The time line corresponding to the run-time execution of the three tasks (corresponding to three SWCs), depicted in Fig. 3, shows multiple outputs corresponding to a single input. The four end-to-end delays are also identified.

Last In First Out (LIFO). This delay is equal to the time elapsed between the current non-overwritten release of task τ_A (input of the data chain) and corresponding first response of task τ_C (output of the data chain).

Last In Last Out (LILO). This delay is equal to the time elapsed between the current non-overwritten release of task τ_A and corresponding last response of task τ_C. This delay is identified as "Data Age"[5] in [21]. Data age specifies the

[5] We will use the term "Data Age delay" to refer to LILO delay throughout the paper.

longest time data is allowed to age from production by the initiator until the data is delivered to the terminator. This delay finds its importance in control applications where the interest lies in the freshness of the produced data. For a data chain in a control system that initiates with a sensor input and terminates by producing an actuation signal, it is very important to ensure that the actuator signal does not exceed a maximum age [21]. Generally speaking, we consider the last non-overwritten input that actually propagates through the data chain towards the output in the case of both LIFO and LILO delays.

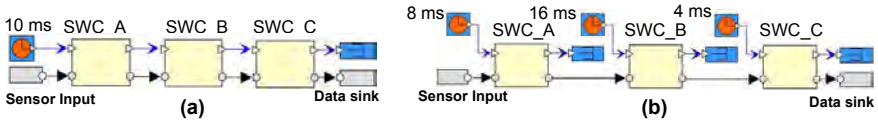


Fig. 2. RCM model of (a) trigger chain (b) data chain in a single-node real-time system

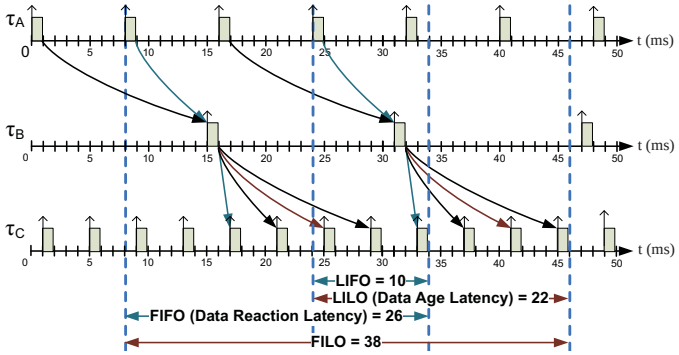


Fig. 3. End-to-end delays for a data chain in a real-time system

First In First Out (FIFO). This delay is equal to the time elapsed between the previous non-overwritten release of task τ_A and first response of task τ_C corresponding to the current non-overwritten release of task τ_A. Assume that a new value of the input is available in the input buffer of task τ_A "just after" the release of the second instance of task τ_A (at time $8ms$). Hence, the second instance of task τ_A "just misses" the read of the new value from its input buffer. This new value has to wait for the next instance of task τ_A to travel towards the output of the data chain. Therefore, the new value will be read by the third and forth instances of task τ_A. The first output corresponding to the new value (arriving just after $8ms$) will appear at the output of the chain at $34ms$. This will result in the FIFO delay of $26ms$ as shown in Fig. 3. This phenomenon is more obvious in the case of distributed embedded systems where a task in the receiving node may just miss to read fresh signals from a message that is received from the network. This delay is identified as "first reaction or Data Reaction"[6] in [21]. It is equal to the longest allowed reaction time for data produced by the initiator to be delivered to the terminator. It finds its importance in button-to-reaction applications in body electronics domain where first reaction to input is important.

[6] We will consistently use the term "Data Reaction delay" to refer to FIFO delay.

First In Last Out (FILO). It is equal to the time elapsed between the previous non-overwritten release of task τ_A and last response of task τ_C corresponding to current non-overwritten release of task τ_A. The reasoning about "just missing" a fresh input (in the case of FIFO delay) is also applicable in this case.

The modeling of data chains and the definition of their end-to-end delays in distributed real-time systems is done in a similar fashion.

3.3. Implemented Holistic Response-Time Analysis

In order to analyze tasks in each node, we implement RTA of tasks with offsets developed by [47, 42] and improved by [29]. We implement the network RTA that supports the analysis of CAN and its high-level protocols. It is based on the following RTA profiles for CAN: (1) RTA of CAN [49, 19]; (2) RTA of CAN for mixed messages [32]; (3) RTA of CAN for mixed messages with offsets [38] (The analysis of this profile is implemented as a standalone analyzer).

The pseudocode of HRTA algorithm is shown in Algorithm 1. The HRTA algorithm iteratively runs the algorithms for node and network analyses. In the first step, release jitter of all messages and tasks in the system is assumed to be zero. The response times of all messages in the network and all tasks in each node are computed. In the second step attribute inheritance is carried out. This means that each message inherits a release jitter equal to the difference between the worst- and best-case response times of its sender task (computed in the first step). Similarly, each task that receives the message inherits a release jitter equal to the difference between the worst- and best-case response times of the message (computed in the first step). In the third step, response times of all messages and tasks are computed again. The newly computed response times are compared with the response times previously computed in the first step. The analysis terminates if the values are equal otherwise these steps are repeated. The conceptual view of HRTA plug-in is shown in Fig. 4.

3.4. Implemented End-to-end Delay Analysis

We implemented the end-to-end delay analysis that is derived in [21] as the E2EDA plug-in for Rubus-ICE. This analysis implicitly requires the calculation of response times of individual tasks, messages and holistic response times of task chains. For example, the calculation of four end-to-end delays for the multi-rate real-time system shown in Fig. 2(b) requires the response time of the task τ_C (corresponding to the component SWC_C) and the activation times of tasks τ_A and τ_C. Since, the HRTA plug-in is able to calculate response times of tasks, network messages and task chains, we reuse the analysis results computed by the HRTA plug-in as an input to the E2EDA plug-in as shown in Fig. 4. The pseudocode of E2EDA algorithm (see [21] for details) is shown in Algorithm 2.

Algorithm 1 Algorithm for holistic response-time analysis

```
1: begin
2: RT_Prev ← 0                                   ▷ Initialize all Response Times (RTs) to zero
3: Repeat ← TRUE
4: while Repeat = TRUE do
5:     for all Messages_and_tasks_in_the_system do
6:         Jitter_Msg ← (WCRT_Sender_task − BCRT_Sender_task)      ▷ WCRT: Worst-Case
   Response Time, BCRT: Best-Case Response Time
7:         Jitter_Receiver_task ← (WCRT_Msg − BCRT_Msg)
8:         COMPUTE_RT_OF_ALL_MESSAGES()
9:         COMPUTE_RT_OF_ALL_TASKS_IN_EVERY_NODE()
10:        if RT > RT_Prev then
11:            RT_Prev ← RT
12:            Repeat ← TRUE
13:        else
14:            Repeat ← FALSE
15:        end if
16:    end for
17: end while
18: end
```

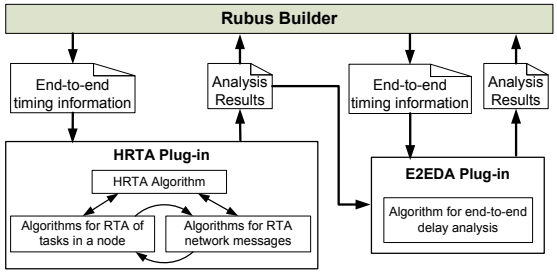


Fig. 4. Conceptual view of the E2EDA plug-in in Rubus-ICE

4. Encountered Problems, Solutions and Experiences

In this section we discuss several problems encountered during the process of implementation and integration of HRTA and E2EDA as plug-ins for the Rubus-ICE tool suite. We also present our solution to each individual problem.

4.1. Extraction of Unambiguous Timing Information

One common assumption in end-to-end response time and delay analyses is that the timing attributes are available as input. However, when these analyses are implemented in a tool chain used for the component-based development of DRE systems, the implementer has to not only code and implement the analysis, but also extract unambiguous timing information from the component model and map it to the inputs for the analysis model. This is because the design and

Algorithm 2 Algorithm for end-to-end delay analysis

begin
GET_RT_OF_ALL_TASKS_MESSAGES_TASK_CHAINS() ▷ Get the analysis results from the HRTA plug-in
FIND_ALL_VALID_TIMED_PATHS() ▷ Timed Path (TP) is a sequence of task instances from input to output. A TP is valid if information flow among tasks is possible [21], e.g., [τ_A(1stinstance), τ_B(1stinstance), τ_C(5thinstance)] in Fig. 3 is a valid TP. On the other hand, TP [τ_A(1stinstance), τ_B(1stinstance), τ_C(1stinstance)] in Fig. 3 is invalid because information cannot flow between τ_B(1stinstance) and τ_C(1stinstance)
procedure COMPUTE_FF_DELAY(FF_TP)
    FF_delay = α_n(instance) + δ_n(instance) - α_1(instance) ▷ α_n(instance): Activation time of the corresponding instance of the nth task in timed path FF_TP
        ▷ δ_n(instance): Response time of the corresponding instance of the nth task in timed path FF_TP
    return FF_delay
end procedure
        ▷ The above mentioned procedure calculates FF$_{Delay}$ only. [21] should be referred for the calculation of the rest of the delays
for all Delay_constraints_specified_in_the_system **do**
    $FF_{Delay} \leftarrow 0, FL_{Delay} \leftarrow 0, LF_{Delay} \leftarrow 0, LL_{Delay} \leftarrow 0$ ▷ Initialize all delays
    COMPUTE_ALL_REACHABLE_TIMED_PATHS() ▷ All those paths from input to output in which the changes in input actually travel towards the output, e.g., [τ_A(2ndinstance), τ_B(1stinstance), τ_C(5thinstance)] in Fig. 3
    $FF\_TP_{count} \leftarrow$ GET_ALL_FF_TPS() ▷ TP: Timed Path, FF: First to First
    $FL\_TP_{count} \leftarrow$ GET_ALL_FL_TPS() ▷ FL: First to Last
    $LF\_TP_{count} \leftarrow$ GET_ALL_LF_TPS() ▷ LF: Last to First
    $LL\_TP_{count} \leftarrow$ GET_ALL_LL_TPS() ▷ LL: Last to Last
    for i:=1 **do**FF_TP_{count}
        if COMPUTE_FF_DELAY(i) $> FF_{Delay}$ **then**
            $FF_{Delay} \leftarrow$ COMPUTE_FF_DELAY()
        end if
    end for
    for i:=1 **do**FL_TP_{count}
        if COMPUTE_FL_DELAY(i) $> FL_{Delay}$ **then**
            $FL_{Delay} \leftarrow$ COMPUTE_FL_DELAY()
        end if
    end for
    for i:=1 **do**LF_TP_{count}
        if COMPUTE_LF_DELAY(i) $> LF_{Delay}$ **then**
            $LF_{Delay} \leftarrow$ COMPUTE_LF_DELAY()
        end if
    end for
    for i:=1 **do**LL_TP_{count}
        if COMPUTE_LL_DELAY(i) $> LL_{Delay}$ **then**
            $LL_{Delay} \leftarrow$ COMPUTE_LL_DELAY()
        end if
    end for
end for
end

analysis models are often build upon different meta-models [22]. Moreover, the design model can contain redundant timing information. Hence, it is not trivial to extract unambiguous timing information for HRTA and E2EDA. The timing information (to be extracted) can be divided into two categories.

Extraction of Timing Information Corresponding to User Inputs. The first category corresponds to the timing attributes of tasks and network messages that are provided in the modeled application by the user. These timing attributes include Worst Case Execution Times (WCETs), periods, minimum update times, offsets, priorities, deadlines, blocking times, precedence relations in task chains, jitters, etc. In [33], we identified all the timing attributes of nodes, networks, transactions, tasks and messages that are required by the HRTA.

Extraction of Timing Information from the Modeled Application. The second category corresponds to the timing attributes that are not directly provided by the user but they must be extracted from the modeled application. For example, message period (in periodic transmission) or message inhibit time (in sporadic transmission) is often not specified by the user. These attributes must be extracted from the modeled application because they are required by the RTA of network communication. In fact, a message inherits the period or inhibit time from the task that queues it. Thus, we assign period or inhibit time to the message which is equal to the period or inhibit time of its sender.

However, the extraction of message timing attributes becomes complex when the sender task has both periodic and sporadic activation patterns. In this case, not only the timing attributes of a message have to be extracted but also the transmission type of the message has to be identified. This problem can be visualized in the example shown in Fig. 5. It should be noted that the Out Software Circuit (OSWC), shown in the figure, is one of the network interface components in RCM that sends a message to the network. Similarly, In Software Circuit (ISWC) receives a message from the network [39].

In Fig. 5(a), the sender task is activated by a clock, and hence the corresponding message is periodic. Similarly, the corresponding message is sporadic in Fig. 5(b) because the sender task is activated by an event. However, the sender task in Fig. 5(c) is triggered by both a clock and an event. Here the relationship between two triggering sources is important. If there exists a dependency relation between them as in the case of mixed transmission in the CANopen protocol [4] and AUTOSAR communication [9] then such message will be treated as a special type of sporadic message. If triggering sources are independent of each other (e.g., in the HCAN protocol [15]), the corresponding message will be considered a mixed message [32, 36]. If there are periodic and sporadic messages in the application, the HRTA plug-in uses the first profile for network analysis (see Section 3.3). On the other hand, if the application contains mixed messages as well, the second profile for network analysis is used.

Identification of Trigger, Data and Mixed Chains. The end-to-end timing requirements on trigger chains are different from those on data chains. These requirements correspond to end-to-end response times for trigger chains and both end-to-end response times and delays for data chains. Data and trigger

chains should be distinctly identified and the corresponding timing requirements should be unambiguously captured in the timing model on which the analysis tools operate. For this purpose, we add a new attribute “trigger dependency” in the data structure of tasks in the analysis model. If a task is triggered by an independent source such as a clock then this attribute will be assigned “independent”. On the other hand, if the task is triggered by another task then this parameter will be assigned “dependent”. Moreover, a precedence constraint will also be specified on this task in the case of dependent triggering.

However, a system can also be modeled with mixed chains that are comprised of data chains as well as trigger chains as shown in Fig. 5(d). In this chain, components SWC_A, SWC_B and SWC_E are triggered by independent clocks and which is the property of components in a data chain. Hence, the “trigger dependency” attribute of the tasks corresponding to these three components will be assigned “independent”. Whereas, the components SWC_C and SWC_D are triggered by their respective predecessors and which is the property of components in a trigger chain. The “trigger dependency” attribute of the tasks corresponding to these two components will be assigned “dependent”.

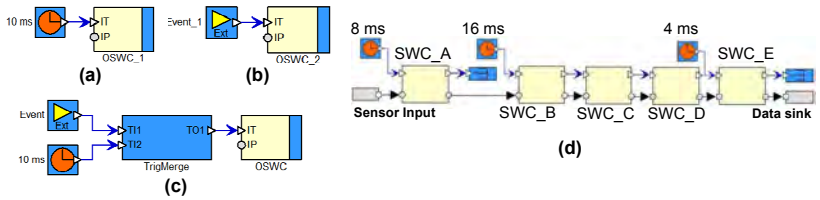


Fig. 5. Extraction of transmission type of a message, (d) RCM model of a mixed chain

4.2. Extraction of Linking Information from Distributed Transactions

In order to perform HRTA, correct linking information of DTs should be extracted from the design model [35]. Consider the following DT in a two-node DRE system shown in Fig. 6. $SWC1 \rightarrow OSWC\_A \rightarrow ISWC\_B \rightarrow SWC2 \rightarrow SWC3$ We identified the need for the following mappings in the component model: between signals and input data ports of OSWCs at the sender node; between signals and the outgoing message at the sender node; between data output ports of ISWC components and the signals (to be sent to the desired components) at the receiver node; between received message and signals at the receiver node; between multiple signals (structure of signals) and a complex data port; and among all trigger ports of network interface components along a DT.

Since, the E2EDA plug-in needs to compute all valid timed paths (i.e., those paths in which input actually travels to the output) from initiator to the terminator for every data chain (see Algorithm 2), the linking information among all tasks and messages in the data chain should be extracted.

4.3. Analysis of Distributed Transactions with Branches

Consider the example of a two-node DRE system containing branches in DTs as shown in Fig. 7. $OSWC\_A1$ and $OSWC\_A2$ in node A send messages $m1$

and $m2$ that are received by $ISWC\_C1$ and $ISWC\_C2$ in node C respectively. Hence, there are two DTs that have different initiators but a single terminator, i.e., SWC_C3 as shown below.
1. $SWC\_A1 \rightarrow SWC\_A2 \rightarrow OSWC\_A1 \rightarrow ISWC\_C1 \rightarrow SWC\_C1 \rightarrow SWC\_C3$
2. $SWC\_A3 \rightarrow OSWC\_A2 \rightarrow ISWC\_C2 \rightarrow SWC\_C2 \rightarrow SWC\_C3$

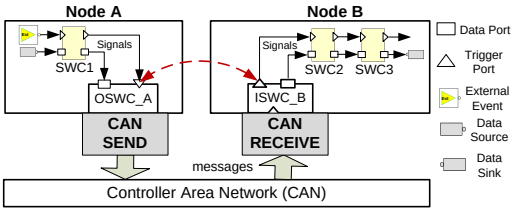


Fig. 6. Two-node DRE system modeled with RCM

Assume that Data Age delay constraint is specified on SWC_C3. Also assume that the start of this constraint is specified on the component SWC_A1 in node A. Therefore, we need to perform end-to-end delay analysis only on the first DT (in the above list). The calculations for Data Age delay require the response time of SWC_C3. However, the response time of this task depends upon the holistic response times of both DTs. In this case, the HRTA plug-in will calculate the holistic response times of all branches whereas the E2EDA plug-in will consider the maximum value among these holistic response times during calculations for the end-to-end delays.

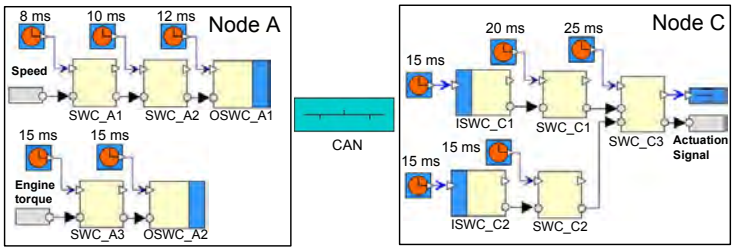


Fig. 7. RCM model of a two-node DRE system with branches in distributed transactions

4.4. Analysis of Mixed Task Chains

There are two options to handle mixed chains in the analysis model. In the first option, if a component is triggered by its predecessor then it is assumed to be triggered by independent clock with the same period as that of its predecessor's clock. Using this option, the execution time line of the task chain corresponding to component chain of Fig. 5(d) is shown in Fig. 8(a). This time line will be used by the E2EDA plug-in to calculate the total number of timed paths. However, there are several timed paths, indicated with crosses in Fig. 8(a), that are impossible to occur in reality. This is because each instance of a task in a trigger chain can be triggered only by one instance of its predecessor task. This will result in unnecessary calculations.

Instead, we use the second option that reduces the number of paths in mixed chain by combining all tasks belonging to a trigger sub-chain into a single task activated by independent clock. Hence, the reduced mixed chain resembles a data chain. For example, SWC_B, SWC_C and SWC_D are combined to a single task (with combined WCETs, offsets, etc.) which is triggered by independent clock whose period is exactly the same as that of the clock that triggers SWC_B component. The execution time line of the task chain corresponding to reduced mixed chain of Fig. 5(d) is shown in Fig. 8(b). The corresponding end-to-end delays are also identified. By implementing the second option , we got rid of the so-called "impossible timed paths". Mixed chains may also exist in the models of DRE systems where they may contain many combinations of data and trigger chains distributed over several nodes. Path reduction in distributed mixed chains is done in a similar fashion.

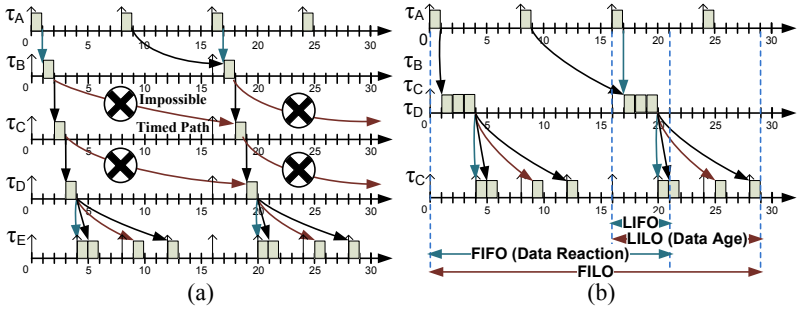


Fig. 8. (a) Impossible timed paths in mixed chains (b) Reduction of a mixed chain

4.5. Analysis of the System Containing "Outside" Messages

One of the requirements by the users of the analysis tools was that the HRTA and E2EDA plug-ins should be able to support the analysis of a system that receives messages from unknown senders (from outside of the modeled application). One motivation behind this requirement may be the integration of two systems that are build using different methodologies and tools. Second motivation could be the integration of legacy systems with newly developed systems. Another motivation could be the requirement for the end-to-end timing analysis early during the development. At early stage, the models of some nodes may not be available. However, the signals and messages which these missing nodes are supposed to send and receive might have been decided. Hence, the network is assumed to contain messages whose sender nodes are not developed yet. Similarly, the available nodes may send messages via network to the nodes that will be available at a later stage.

The HRTA connects the tasks and messages in a DT by means of attribute inheritance [48]. Moreover, the message also inherits other attributes from the sender task such as transmission type (periodic, sporadic or mixed [32]); and period or inhibit time or both. The only problem with this requirement is that a message, obviously, cannot inherit these attributes if the sender is unknown or the message is received from outside of the model. In order to solve this

problem, each such message is assumed to be the initiator of the corresponding DT. The transmission type and period (or inhibit time or both) of this message are extracted from the user input (instead of the sending task as in the case of intra-model messages). However, the forward attribute inheritance is valid, i.e., the receiver task will inherit the difference between the worst- and best-case response times of the message as its release jitter.

4.6. Impact of Component Technology on the Analysis Implementation

The design decisions made in the component technology (i.e., RCM) can have indirect impact on the response times computed by the analysis. For example, design decisions could have impact on WCETs and blocking times which in turn have impact on the response times. In order to implement, integrate and test HRTA and E2EDA, the implementer needs to understand the design model (component technology), analysis model and run-time translation of the design model. In the design model, the architecture of an application is described in terms of software components, their interconnections and software architectures. Whereas in the analysis model, the application is defined in terms of tasks, transactions, messages and timing parameters. At run-time, a task may correspond to a single component or a chain of components. The run-time translation of a component may differ among different component technologies.

4.7. Direct Cycles in Distributed Transactions

A direct cycle in a DT is formed when any two tasks located on different nodes send messages to each other. When there are direct cycles in a DT, the HRTA may run forever (if deadlines are not specified) because the response times increase in every iteration. Consider a two-node application modeled in RCM as shown in Fig. 9 (a). The $OSWC\_A$ component in node A sends a message $m1$ to node B where it is received by $ISWC\_B$. Similarly, $OSWC\_B$ in node B sends a message $m2$ to $ISWC\_A$ in node A.

There are two options for the run-time allocation of network interface component (OSWC or ISWC) as shown in Fig. 9 (b). First option is to allocate it to the task that corresponds to the immediate SWC, i.e., the component that receives/sends the signals from/to it. Since SWC_A is immediately connected to both network interface components in node A, there will be only one task in node A denoted by τ_A as shown in Fig. 9 (b). Similarly, τ_B is the run-time representation of $ISWC\_B$, SWC_B and $OSWC\_B$. Obviously, this run-time allocation will result in direct cycles. This problem may appear in those component technologies which do not use exclusive modeling objects or means to differentiate between intra- and inter-node communication in the design model and rely completely on the run-time environment to handle the communication. Hence, some special methods are required to avoid direct cycles in these technologies.

The direct cycles can be avoided by allocating each network interface component to a separate task as shown in the option 2 in Fig. 9 (b). Although same messages are sent between the nodes, one task cannot be both a sender and

a receiver. No doubt, there is a cycle between the nodes, but not a direct one. Hence, the HRTA may produce converging results, and non-terminating execution of the plug-in may be avoided. It is interesting to note that the requirements and limitations of the analysis implementation may provide feedback to the design decisions concerning the run-time allocation of modeling components.

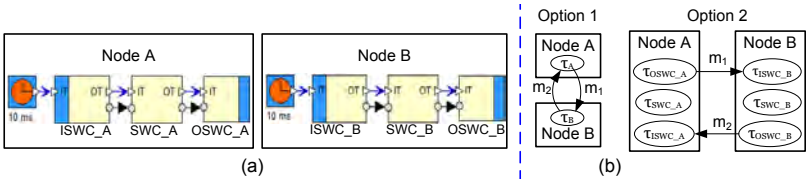


Fig. 9. Options for the run-time allocation of network interface components

4.8. Sequential Execution of Plug-ins in Rubus Plug-in Framework

The plug-in framework in Rubus-ICE allows only sequential execution of plug-ins. There exists a plug-in in Rubus-ICE that can perform RTA of tasks in a node and it is already in the industrial use. There are two options to develop the HRTA plug-in for Rubus-ICE as shown in Fig. 10. The option A supports reusability by building the HRTA plug-in by integrating existing RTA plug-in with two new plug-ins, i.e., one implementing network RTA and the other implementing the HRTA. In this case, the HRTA plug-in will be lightweight. It iteratively uses the analysis results produced by the node and the network RTA plug-ins and accordingly provides new inputs to them until converging holistic response times are obtained or the deadlines (if specified) are violated. On the other hand, option B requires the development of the HRTA plug-in from the scratch, i.e, implementing the algorithms of node, network and the HRTA. This option does not support any reuse of existing plug-ins.

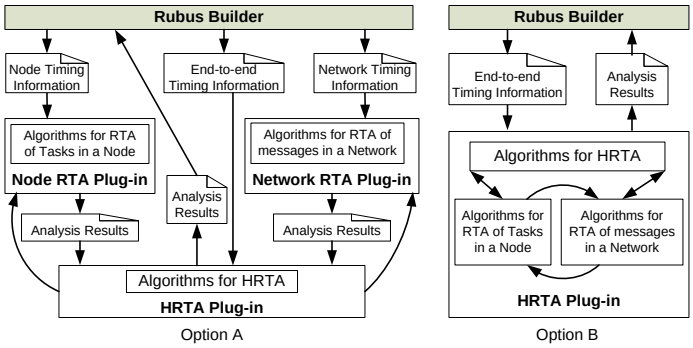


Fig. 10. Options to develop the HRTA Plug-in for Rubus-ICE

Since, option A allows the reuse of a pre-tested and heavyweight node RTA plug-in, it is easy to implement and requires less time for implementation, integration and test compared to option B. However, the implementation method in option A is not supported by the plug-in framework of Rubus-ICE because the plug-ins can only be executed sequentially. Hence, we selected option B for the

implementation of the HRTA. Since E2EDA algorithm is non-iterative, there is no need to build the E2EDA plug-in from the scratch. In fact, the HRTA plug-in can be completely reused as a black box. This means that the response times of tasks, messages and task chains computed by the HRTA plug-in can be used as one of the inputs for the E2EDA plug-in as shown in Fig. 4.

4.9. Analysis of DRE Systems with Multiple Networks

In a DRE system, a node may be connected to more than one network. This type of node is called a gateway node. If a transaction is distributed over more than one network, the computation of its holistic response time involves the analysis of more than one network. Such transaction is divided into sub- transactions (each having a single network) which are analyzed separately in the first step. In the second step, the attribute inheritance is carried out (see Section 3.3) and the sub-transactions are analyzed again. The second step is repeated until the response times converge or the deadlines (if specified) are violated. Although, we analyze the sub-transactions separately, the multi-step analysis (especially attribute inheritance step) makes the overall analysis to be holistic. The implemented HRTA does not support the analysis of a transaction that is distributed cyclically on multiple networks, i.e., the transactions that is distributed over more than one network while its first and last tasks are located on the same network. Since, the E2EDA plug-in receives the response times from the HRTA plug-in, it does not need to split the system into sub-systems.

4.10. Specification of Delay Constraints on Data Paths

One issue that concerns both modeling and analysis is how to specify the delay constraints on data paths in both data and mixed chains. This is important because the delay constraints specified in the modeled application have to be extracted in the timing model and the end-to-end delays have to be computed only for the specified data path(s) by the E2EDA plug-in. For this purpose, we introduce start and end objects for each of the four delay constraints (discussed in Subsection 3.2) in the component technology. The constraint object has a meaningful name, and start and end points along a data path. Fig. 11 shows the “Data Age” delay constraint specified on a sensor-actuator data path. Similarly, there are start and end objects for “Data Reaction”, “LIFO” and “FILO” delays. A delay constraint can also be distributed over several nodes. Another useful method for specifying the delay constraints is by selecting each component (e.g., with mouse click) along the data path.

4.11. Presentation of Analysis Results

When HRTA of a modeled application has been performed, the next issue is how to present the analysis results. There can be a large number of tasks and messages in the system. It may not be appropriate to display the response

times of all tasks, messages and DTs in the system because it may contain a lot of useless information (if the user is not interested in all of it). A way around this problem is to provide the end-to-end response times and delays of only those tasks and DTs which have deadline requirements and delay constraints (specified by the user) or which produce control signals for external actuators. Apart from this, we also provide an option for the user to get detailed analysis results from both the HRTA and E2EDA plug-ins. The analysis report also shows network utilization which is defined as the sum of the ratio of transmission time to the corresponding period (or minimum-update time) for all messages [32].

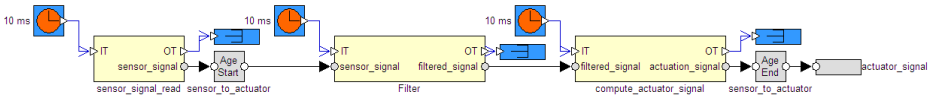


Fig. 11. Age delay constraint specified on a data path

4.12. Interaction between the User and the HRTA Plug-in

We feel that it is important to display the number of iterations, running time and over all progress of the plug-in during its execution. Further, the user should be able to interact with the plug-in, i.e., stop, rerun or exit the plug-in at any time.

4.13. Suggestions to Improve Schedulability Based on Analysis Results

If the analysis results indicate that the modeled system is unschedulable, it can be interesting if the HRTA plug-in is able to provide suggestions (e.g., by varying system parameters) guiding the user to make the system schedulable. However, it is not trivial to provide such feedback because there can be so many reasons behind the system being not schedulable. Another interesting and related feature would be to provide a trace analyzer as another plug-in that can be used after system has been developed. This analyzer will record the execution of the actual system and then present a graphical comparison of the trace with response times of tasks and messages; holistic response times of trigger, data and mixed chains; and end-to-end delays of data and mixed chains. Based on such comparisons, the user may have better understanding of how the schedulability of the system can be improved. The support for this type of feedback in the HRTA plug-in will be provided in the future.

4.14. Continuous Collaboration between Integrator and Implementer

Our experience shows that there is a need for continuous collaboration between the integrator of the plug-ins and its implementer especially during the phase of integration testing. This collaboration is more obvious when the plug-in is developed in isolation by the implementer (from research background) and integrated with the industrial tool chain by the integrator (with limited experience of integrating complex real-time analysis but aware of overall objective). A continuous consultation and communication was required between the integrator

and the implementer for the verification of the plug-ins. Examples of small DRE systems with varying architectures were created for the verification. The implementer had to verify these examples by hand. The integration testing and verification of the HRTA plug-in was non-trivial and most tedious activity.

5. Automotive Application Case Study

We provide a proof of concept for the analyses that we implemented in the Rubus-ICE by conducting the automotive-application case study. First, we model Autonomous Cruise Control (ACC) system with RCM using Rubus-ICE. Then, we analyze the modeled ACC system using the HRTA and E2EDA plug-ins.

5.1. Autonomous Cruise Control System

A Cruise Control (CC) system is an automotive feature that allows a vehicle to automatically maintain a steady speed to the value that is preset by the driver. It uses velocity feedback from the speed sensor (e.g., a speedometer) and accordingly controls the engine throttle. However, it does not take into account traffic conditions around the vehicle. Whereas, an Autonomous Cruise Control (ACC) system allows the CC of the vehicle to adapt itself to the traffic environment without communicating (cooperating) with the surrounding vehicles. Often, it uses a radar to create a feedback of distance to and velocity of the preceding vehicle. Based on the feedback, it either reduces the vehicle speed to keep a safe distance and time gap from the preceding vehicle or accelerates the vehicle to match the preset speed specified by the driver [41]. The ACC system may be divided into four subsystems, i.e., Cruise Control (CC), Engine Control (EC), Brake Control (BC) and User Interface (UI) [14] as shown in Fig. 12. The subsystems communicate with each other via the CAN network.

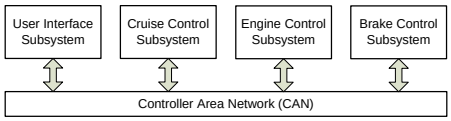


Fig. 12. Block diagram of Autonomous Cruise Control System

User Interface (UI) Subsystem. It reads inputs (provided by the driver) and shows status messages and warnings on the display screen. The inputs are acquired by means of switches and buttons mounted on the steering wheel. These include Cruise Switch input that corresponds to ON/OFF, Standby and Resume (resuming to a speed predefined by the driver) states for ACC; Set Speed input (desired cruising speed set by the driver) and desired clearing distance from the preceding vehicle. It also receives messages that include linear and angular speed of the vehicle, status of manual brake sensor, state of ACC subsystem, status messages and warnings to be displayed on the screen. It also sends messages (including status of driver's input) to other subsystems.
Cruise Control (CC) Subsystem. The CC subsystem receives user input information as a CAN message from the UI subsystem. From the received message it analyzes the state of the CC switch; if it is in ON state then it activates

the CC functionality. It reads input from the proximity sensor (e.g., radar) and processes it to determine the presence of a vehicle in front of it. Moreover, it processes the radar signals along with the information received from other subsystems such as vehicle speed to determine its distance from the preceding vehicle. Accordingly, it sends control information to the BC and EC subsystems to adjust the speed of the vehicle with the cruising speed or clearing distance from the preceding vehicle. It also receives the status of manual brake sensor from the BC subsystem. If brakes are pressed manually then the CC functionality is disabled. It also sends status messages to the UI subsystem.

Engine Control (EC) Subsystem. The EC subsystem is responsible for controlling the vehicle speed by adjusting engine throttle. It reads sensor input and accordingly determines engine torque. It receives CAN messages from other subsystems that include information regarding vehicle speed, status of manual brake sensor, and input information processed by the UI system. Based on this information, it determines whether to increase or decrease engine throttle. It then sends new throttle position to the actuators that control engine throttle.

Brake Control (BC) Subsystem. The BC subsystem receives inputs from sensor for manual brakes status and linear and angular speed sensors connected to all wheels. It also receives a CAN message that includes control information processed by the CC subsystem. Based on this feedback, it computes new vehicle speed. Accordingly, it produces control signals and sends them to the brake actuators and brake light controllers. It also sends CAN messages to other subsystems that carry status of manual brake, vehicle speed and RPM.

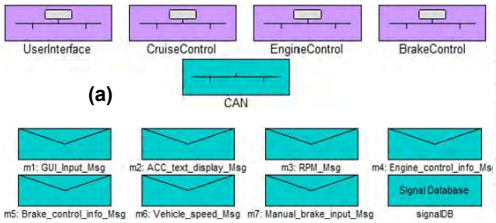
5.2. Modeling of ACC System with RCM in Rubus-ICE

In RCM, we model each subsystem as a separate node connected to a CAN network as shown in Fig. 13(a). The selected speed of the CAN bus is 500 kbps. The extended frame format is selected, i.e., each frame will use 29-bit identifier [24]. The ACC system is modeled with trigger, data and mixed chains.

There are seven CAN messages in the system as shown in Fig. 13(b). A signal data base "signalDB" that contains all the signals sent to the network is also shown. Each signal in the signalDB is linked to one or more messages. The extracted attributes of all messages including data size (s_m), priority (P_m), transmission type (ξ_m) and period or inhibit time (T_m) are listed in the table shown in Fig. 13(c). The high-level architectures of CC, EC, BC and UI nodes modeled with RCM are shown in Fig. 14(a), 14(b), 14(c) and 14(d) respectively.

Internal Model of CC Node in RCM. The CC node is modeled with four assemblies as shown in Fig. 14(a). An assembly in RCM is a container for various software items. The Input_from_Sensors assembly contains one SWC that reads radar sensor values as shown in Fig. 15. The Input_from_CAN assembly contains three ISWCs, i.e., GUI_Input_Msg_ISWC, Vehicle_speed_Msg_ISWC and Manual_brake_input_Msg_ISWC as depicted in Fig. 16(a). These components receive messages $m1$, $m6$ and $m7$ from the CAN respectively. The assembly Output_to_CAN contains three OSWC components that send messages $m5$,

$m4$ and $m2$ to the CAN network as shown in Fig. 16(b). The Cruise_Control assembly contains two SWCs: one handles the input and CC mode signals while the other processes the received information and produces control messages for the other nodes. The internal model of this assembly is shown in Fig. 17.



Msg	s_m	P_m	ξ_m	T_m (μSec)	C_m (μSec)
$m1$	8	7	Periodic	10000	320
$m2$	8	6	Periodic	10000	320
$m3$	8	4	Periodic	10000	200
$m4$	8	3	Sporadic	10000	320
$m5$	2	5	Sporadic	10000	320
$m6$	2	2	Periodic	10000	200
$m7$	1	1	Sporadic	10000	180

(c)

Fig. 13. (a) RCM model of ACC system, (b) RCM model of CAN messages and signal database, (c) message attributes extracted from the model

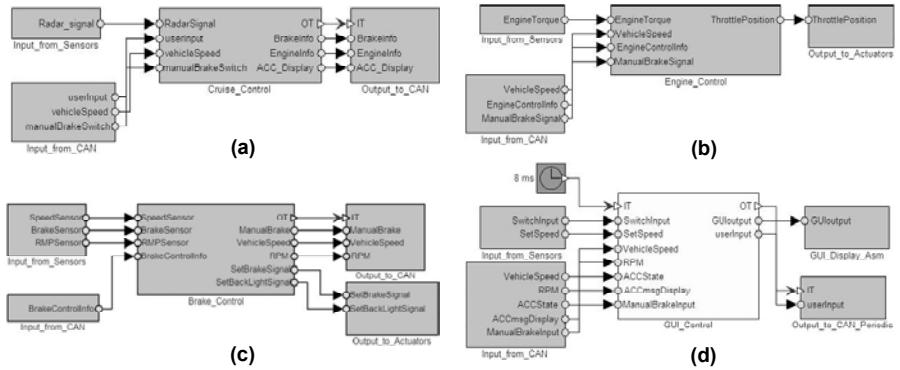


Fig. 14. RCM model of (a) CC node, (b) EC node, (c) BC node, (d) UI node

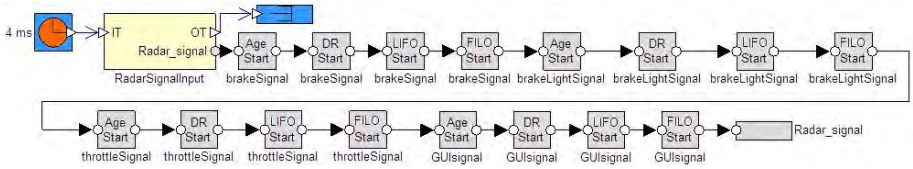


Fig. 15. CC node: Internal model of the Input_from_Sensors assembly

Internal Model of EC Node in RCM. The EC node is modeled with four assemblies as shown in Fig. 14(b). The Input_from_Sensors assembly contains one SWC that reads the sensor values corresponding to the engine torque as shown in Fig. 18(a). The Input_from_CAN assembly contains three ISWCs, i.e., Vehicle_Speed_Msg_ISWC, Engine_control_info_Msg _ISWC and Manual_brake_input _Msg_ISWC as shown in Fig. 18(b). These components receive messages $m6$, $m4$ and $m7$ from the CAN network respectively. The third assembly, Output_to_Actuators, shown in Fig. 18(c), contains the SWC that produces control signals for the engine throttle actuator. The fourth assembly Engine_Control, shown in Fig. 19, contains two SWCs: one handles and processes the inputs from sensors and received messages, while the other computes the new position for the

engine throttle. These components are part of a distributed mixed chain that we will analyze along with other distributed mixed chains in the next subsections.

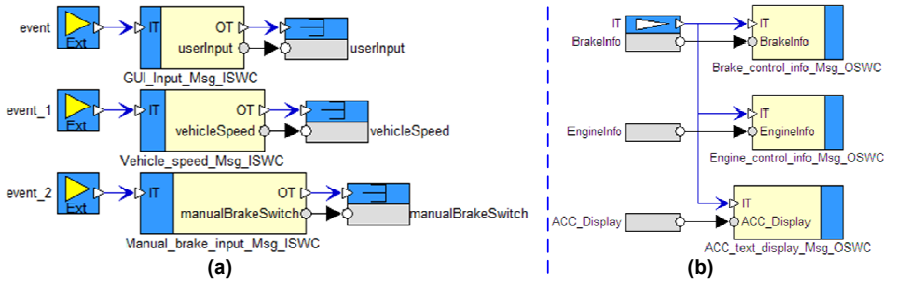


Fig. 16. CC node: Internal model of assemblies (a) Input_from_CAN, (b) Output_to_CAN

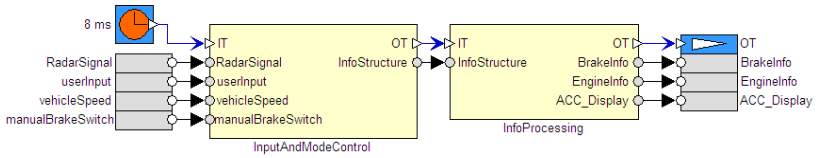


Fig. 17. CC node: SWCs comprising the Cruise_Control assembly

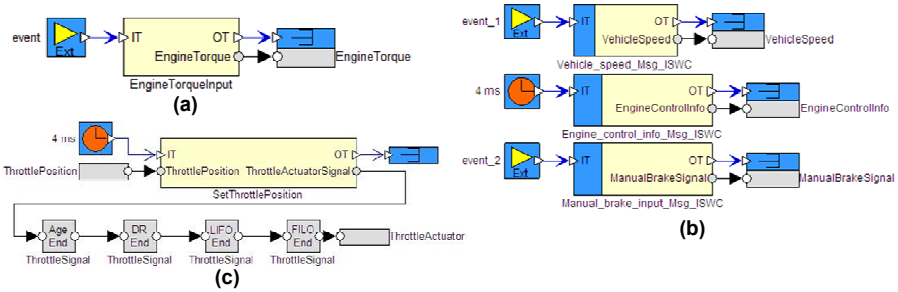


Fig. 18. EC node: Internal model of assemblies (a) Input_from_Sensors, (b) Input_from_CAN, (c) Output_to_Actuators

Internal Model of BC Node in RCM. The BC node is modeled with five assemblies as shown in Fig. 14(c). The Input_from_Sensors assembly contains three SWCs as shown in Fig. 20(a). These SWCs read the sensor values that correspond to the values of speed, rpm and manual brake sensors in the vehicle. The Input_from_CAN assembly, shown in Fig. 20(b), contains the ISWC component Brake_control_info_Msg_ISWC that receives a message $m5$ from the CAN. The third assembly, i.e., Brake_Control as shown in Fig. 21(a), contains two SWCs: one handles and processes the inputs from sensors and received messages while the other computes the control signals for brake actuators. The fourth assembly Output_to_CAN contains three OSWC components as shown in Fig. 20(c). These components send messages $m7$, $m6$ and $m3$ to the CAN. The fifth assembly, Output_to_Actuators as shown in Fig. 21(b), contains the SWCs that produce control signals for the brake actuators and brake light controllers.

Internal Model of UI Node in RCM. The UI node is modeled with four assemblies along with one SWC as shown in Fig. 14(d). The GUI_Control SWC handles the input from the sensors and messages from the CAN. After processing the information, it not only produces information for Graphical User

Interface (GUI), but also computes control signals for the other nodes. The Input_from_Sensors assembly contains two SWCs as shown in Fig. 22(a). One of them reads the sensor values that correspond to the state of the cruise control switch on the steering wheel. The other SWC reads the sensor values that correspond to the vehicle cruising speed set by the driver. The Input_from_CAN assembly contains four ISWC components, i.e., Vehicle_Speed_Msg_ISWC, RPM_Msg_ISWC, Manual_brake_input_Msg_ISWC and ACC_text_display_Msg_ISWC as shown in Fig. 22(b). These components receive messages $m6$, $m3$, $m7$ and $m2$ from the CAN respectively. The third assembly, i.e., Output_to_ CAN_Periodic sends a message $m1$ to the CAN via the OSWC component as shown in Fig. 22(c). The fourth assembly, i.e., GUI_Display_Asm contains one SWC, i.e., GUIdisplay component as shown in Fig. 23. This component sends the signals (corresponding to updated information) to GUI in the car.

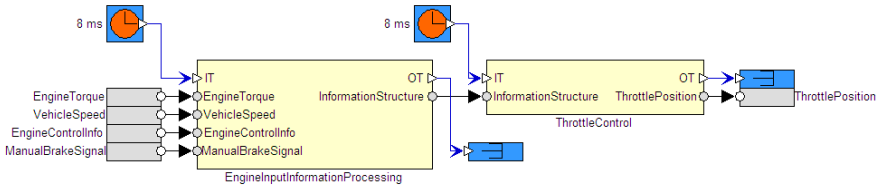


Fig. 19. EC node: SWCs comprising the Engine_Control assembly

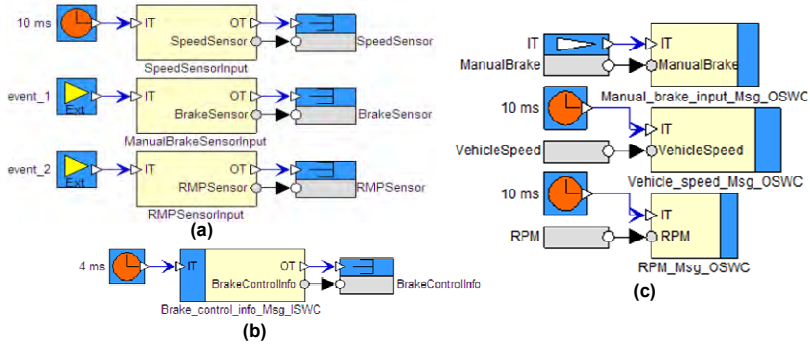


Fig. 20. BC node: Internal model of assemblies (a) Input_from_Sensors, (b) Input_from_CAN, (c) Output_to_CAN

5.3. Modeling of End-to-end Deadline Requirements

We specify end-to-end deadline requirements on four DTs in the ACC system using the deadline object in RCM. All these DTs, i.e., DT_1, DT_2, DT_3 and DT_4 are distributed mixed chains as shown in Table 1. All these chains have one common initiator, i.e., their first task corresponds to the SWC that reads radar signal which is denoted by *RadarSignalInput* and located in the CC node as shown in Fig. 15. The last tasks of DT_1 and DT_2 are located in the BC node. These tasks correspond to the SWCs *SetBrakeSignal* and *SetBrakeLightSignal* as shown in Fig. 14(c). These two tasks are responsible for producing brake actuation and brake light control signals respectively. The last task of DT_3 corresponds to *SetThrottlePosition* SWC and is located in the EC node as shown

in Fig. 14(b). It produces control signal for the engine throttle actuator. The last task of DT_4 corresponds to *GUIdisplay* SWC and is located in the UI node as shown in Fig. 14(d). This task provides display information for the driver.

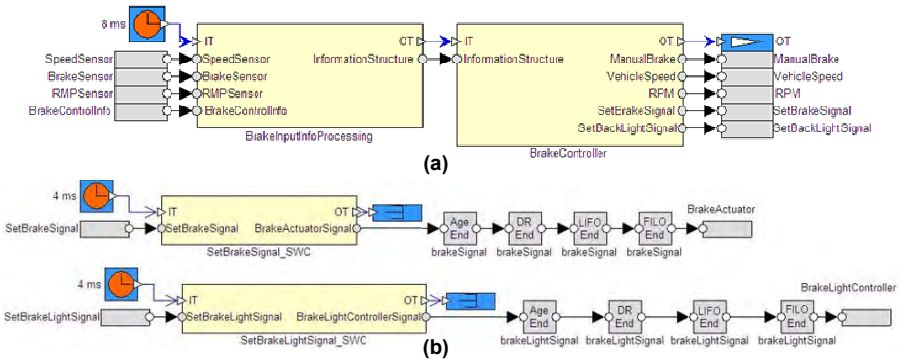


Fig. 21. BC node: Model of assemblies (a) Brake_Control (b) Output_to_Actuators

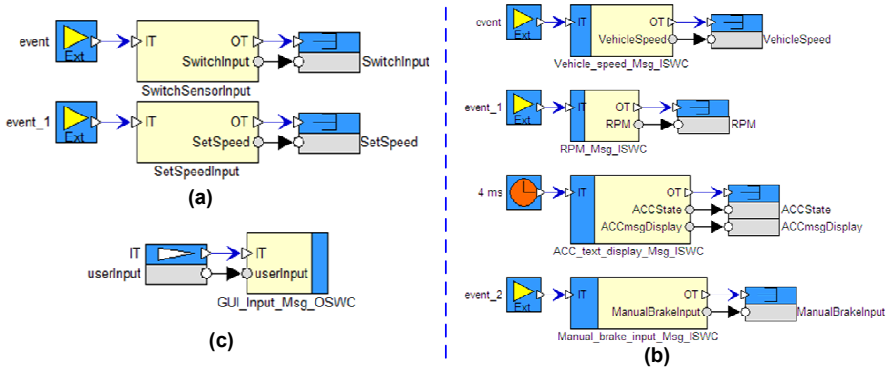


Fig. 22. UI node: Internal model of assemblies (a) Input_from_Sensors, (b) Input_from_CAN, (c) Output_to_CAN_Periodic

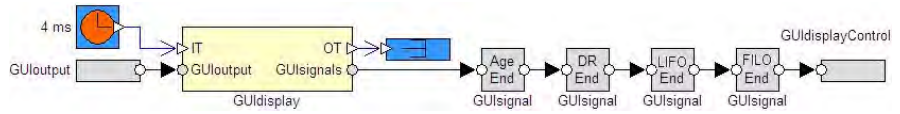


Fig. 23. UI node: Internal model of the GUI_Display_Asm assembly

All the mixed chains under analysis are distributed over more than one node. We list all the components in the data path (from initiator to terminator) of each chain as shown below. We also specify four delay constraints (discussed in Section 3) on each DT under analysis. In RCM, the model of each delay constraint consists of start object and end object. The start objects for all four delay constraints for each DT are shown in Fig. 15. There are sixteen start objects for delay constraints in Fig. 15 because there are four DTs under analysis with four delay constraints specified on each. The end objects for all delay constraints for DT_1 and DT_2 are specified in Fig. 21(b). Similarly, the end objects for all delay constraints for DT_3 and DT_4 are specified in Fig. 18(c) and Fig. 23 respectively.

1. DT_1: $RadarSignalInput \rightarrow InputAndModeControl \rightarrow InfoProcessing \rightarrow Brake\_control\_info\_Msg\_OSWC \rightarrow message : m5 \rightarrow Brake\_control\_info\_Msg\_ISWC \rightarrow BrakeInputInfoProcessing \rightarrow BrakeController \rightarrow SetBrakeSignal\_SWC$
2. DT_2: $RadarSignalInput \rightarrow InputAndModeControl \rightarrow InfoProcessing \rightarrow Brake\_control\_info\_Msg\_OSWC \rightarrow message : m5 \rightarrow Brake\_control\_info\_Msg\_ISWC \rightarrow BrakeInputInfoProcessing \rightarrow BrakeController \rightarrow SetBrakeLightSignal\_SWC$
3. DT_3: $RadarSignalInput \rightarrow InputAndModeControl \rightarrow InfoProcessing \rightarrow Engine\_control\_info\_Msg\_OSWC \rightarrow message : m4 \rightarrow Engine\_control\_info\_Msg\_ISWC \rightarrow EngineInputInformationProcessing \rightarrow ThrottleControl \rightarrow SetThrottlePosition$
4. DT_4: $RadarSignalInput \rightarrow InputAndModeControl \rightarrow InfoProcessing \rightarrow ACC\_text\_display\_Msg\_OSWC \rightarrow message : m2 \rightarrow ACC\_text\_display\_Msg\_ISWC \rightarrow GUI\_Control \rightarrow GUIdisplay$

5.4. Analysis of ACC System using the HRTA and E2EDA Plug-ins

The run-time allocation of all the components in the model of the ACC system results in 19 transactions, 36 tasks and 7 messages. We provide the analysis results of only those transactions on which deadline requirements or delay constraints are specified. The transmission times (C_m) of all messages computed by the HRTA plug-in are listed in the table shown in Fig. 13(c). The WCET of each component in the modeled ACC system is selected from the range of 10-60 μSec. The HRTA plug-in analyzes all four DTs (discussed in the previous subsection). Once the HRTA plug-in has completed its execution and produced analysis results then the E2EDA plug-in analyzes only those DTs on which end-to-end delay constraints are specified (i.e., all four DTs).

The analysis report in Table 1 provides worst-case holistic response times of the four distributed mixed chains using the HRTA plug-in. The corresponding deadlines are also shown. The response time of a DT is counted from the activation of the first task to the completion of the last task in the chain. The response times of these four DTs correspond to the production of control signals for brake actuators, brake lights controllers, engine throttle actuator and GUI. The analysis report produced by the E2EDA plug-in is shown in Table 2. It lists four end-to-end delays calculated for each DT. The corresponding specified delay constraints are also listed in the table. By comparing the end-to-end deadlines and specified delay constraints with the calculated holistic response times and end-to-end delays in Tables 1 and 2 respectively, we see that the modeled ACC system meets all of its deadlines.

6. Conclusion and Future Work

We presented the implementation of the state-of-the-art Holistic Response Time Analysis (HRTA) and End-to-End Delay Analysis (E2EDA) as two individual plug-ins for the existing industrial tool suite Rubus-ICE. The implemented analyses are general as they support the integration of real-time analysis of various

networks without a need for changing the end-to-end analysis algorithms. With the implementation of these plug-ins, Rubus-ICE is able to support distributed end-to-end timing analysis of trigger flows as well as asynchronous data flows which are common in automotive embedded systems.

Table 1. Analysis report by the HRTA plug-in

Distributed Transaction	Chain Type	Control Signal Produced by the Chain	Deadline (μSec)	Holistic Response Time (μSec)
DT_1	Mixed Chain	SetBrakeSignal	1000	220
DT_2	Mixed Chain	SetBrakeLightSignal	1000	280
DT_3	Mixed Chain	SetThrottlePosition	1000	130
DT_4	Mixed Chain	GUIdisplay	1500	345

Table 2. Analysis report by the E2EDA plug-in

Distributed Transaction	DT_1	DT_2	DT_3	DT_4
Specified Age Delay Constraint(μSec)	5000	5000	5000	5000
Calculated Age Delay (μSec)	4220	4280	4130	4345
Specified Reaction Delay Constraint(μSec)	10000	10000	10000	10000
Calculated Reaction Delay (μSec)	8220	8280	8130	8345
Specified LIFO Delay Constraint(μSec)	1000	1000	1000	1500
Calculated LIFO Delay (μSec)	220	280	130	345
Specified FILO Delay Constraint(μSec)	15000	15000	15000	15000
Calculated FILO Delay (μSec)	12220	12280	12130	12345

There are many challenges faced by the implementer when state-of-the-art real-time analyses like HRTA and E2EDA are transferred to the industrial tools. The implementer has to not only code and implement the analyses in the tools, but also deal with various challenging issues in an effective way with respect to time and cost. We discussed and solved several issues that we faced during the implementation, integration and evaluation of the plug-ins. The experience gained by dealing with the implementation challenges provided a feed back to the component technology. We found the integration testing to be a tedious and non-trivial activity. Our experience of implementing, integrating and evaluating these plug-ins shows that a considerable amount of work and time is required to transfer complex real-time analysis results to the industrial tools.

We provided a proof of concept by modeling the ACC system with component-based approach using the existing industrial component model (Rubus Component Model) and analyzing it with the HRTA and E2EDA plug-ins.

We believe that most of the problems discussed in this paper are generally applicable when real-time analysis is transferred to any industrial or academic tool suite. The contributions in this paper may provide guidance for the implementation of other complex real-time analysis techniques in any industrial tool suite that supports plug-in framework for the integration of new tools and allows component-based development of distributed real-time embedded systems.

In the future, we plan to implement the analysis of other network communication protocols (e.g., Flexray, switched ethernet, etc.) and integrate them within the HRTA plug-in. Another future work is the implementation of RTA for CAN

with FIFO and work-conserving queues [18, 20], and RTA of CAN with FIFO Queues for Mixed Messages [36] within HRTA plug-in. We also plan to integrate the stand alone analyzer, that we developed for the analysis of mixed messages with offsets [38], with the HRTA plug-in.

References

1. Arcticus Systems, http://www.arcticus-systems.com
2. BAE Systems Hägglunds, http://www.baesystems.com/hagglunds
3. CANoe. www.vector.com/portal/medien/cmc/info/canoe_productinformation_en.pdf
4. CANopen Application Layer and Communication Profile. CiA Draft Standard 301. Version 4.02. February 13, 2002, http://www.can-cia.org/index.php?id=440
5. Knorr-bremse, web page, http://www.knorr-bremse.com
6. MAST–Modeling and Analysis Suite for RT Applications, http://mast.unican.es
7. Mecel, web page, http://www.mecel.se
8. RAPID RMA: The Art of Modeling Real-Time Systems, www.tripac.com/rapid-rma
9. Requirements on Communication, Release 3.0, Revision 7, Ver. 2.2.0. The AUTOSAR Consortium, September, 2010, www.autosar.org
10. The Volcano Family, http://www.mentor.com/products/vnd
11. Vector. http://www.vector.com
12. Volcano Network Architect. Mentor Graphics, http://www.mentor.com/products/vnd/communication-management/vna
13. Volvo Construction Equipment, http://www.volvoce.com
14. Adaptive Cruise Control System Overview. In: Workshop of Software System Safety Working Group (April 2005), Anaheim, California, USA
15. Hägglunds Controller Area Network (HCAN), Network Implementation Specification. BAE Systems Hägglunds, Sweden (internal document) (April 2009)
16. TIMMO Methodology , Version 2. TIMMO (TIMing MOdel), Deliverable 7 (Oct 2009)
17. Audsley, N., Burns, A., Davis, R., Tindell, K., Wellings, A.: Fixed priority pre-emptive scheduling:an historic perspective. Real-Time Systems 8(2/3), 173–198 (1995)
18. Davis, R., Navet, N.: Controller Area Network (CAN) Schedulability Analysis for Messages with Arbitrary Deadlines in FIFO and Work-Conserving Queues. In: 9th IEEE International Workshop on Factory Communication Systems. pp. 33 –42 (May 2012)
19. Davis, R., Burns, A., Bril, R., Lukkien, J.: Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. RTS 35, 239–272 (2007)
20. Davis, R.I., Kollmann, S., Pollex, V., Slomka, F.: Controller Area Network (CAN) Schedulability Analysis with FIFO queues. In: ECRTS 2011
21. Feiertag, N., Richter, K., Nordlander, J., Jonsson, J.: A Compositional Framework for End-to-End Path Delay Calculation of Automotive Systems under Different Path Semantics. In: CRTS, 2008
22. Hagner, M., Goltz, U.: Integration of scheduling analysis into uml based development processes through model transformation. In: International Multi-conference on Computer Science and Information Technology (IMCSIT). pp. 797 –804 (Oct 2010)
23. Hamann, A., Henia, R., Racu, R., Jersak, M., Richter, K., Ernst, R.: Symta/s - symbolic timing analysis for systems (2004)
24. ISO 11898-1: Road Vehicles interchange of digital information controller area network (CAN) for high-speed communication, ISO Standard-11898, Nov. 1993.
25. Joseph, M., Pandya, P.: Finding Response Times in a Real-Time System. The Computer Journal (British Computer Society) 29(5), 390–395 (October 1986)

26. K. Hänninen et.al.: Framework for real-time analysis in Rubus-ICE. In: 13th IEEE Conference on Emerging Technologies and Factory Automation) (2008)
27. K. Hänninen et.al.: The Rubus Component Model for Resource Constrained Real-Time Systems. In: 3rd IEEE Symposium on Industrial Embedded Systems (2008)
28. Liu, C., Layland, J.: Scheduling algorithms for multi-programming in a hard-real-time environment. ACM 20(1), 46–61 (1973)
29. Mäki-Turja, J., , Nolin, M.: Tighter response-times for tasks with offsets. In: Real-time and Embedded Computing Systems and Applications Conference (August 2004)
30. Mubeen, S.: Modeling and timing analysis of industrial component-based distributed real-time embedded systems. Licentiate thesis, Mälardalen University (January 2012), http://www.mrtc.mdh.se/index.php?choice=publications&id=2748
31. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending response-time analysis of controller area network (CAN) with FIFO queues for mixed messages. In: 16th IEEE Conference on Emerging Technologies and Factory Automation (September 2011)
32. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending schedulability analysis of controller area network (CAN) for mixed (periodic/sporadic) messages. In: 16th IEEE Conference on Emerging Technologies and Factory Automation (ETFA) (September 2011)
33. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extraction of end-to-end timing model from component-based distributed real-time embedded systems. In: Time Analysis and Model-Based Design, from Functional Models to Distributed Deployments (TiMoBD) workshop located at Embedded Systems Week. pp. 1–6. Springer (October 2011)
34. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Implementation of Holistic Response-Time Analysis in Rubus-ICE: Preliminary Findings, Issues and Experiences. In: The 32nd IEEE Real-Time Systems Symposium, WIP Session. pp. 9–12 (December 2011)
35. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Tracing event chains for holistic response-time analysis of component-based distributed real-time systems. SIGBED Review 8, 48–51 (September 2011), http://doi.acm.org/10.1145/2038617.2038628
36. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Response-Time Analysis of Mixed Messages in Controller Area Network with Priority- and FIFO-Queued Nodes. In: 9th IEEE International Workshop on Factory Communication Systems (WFCS) (May 2012)
37. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Support for Holistic Response-time Analysis in an Industrial Tool Suite: Implementation Issues, Experiences and a Case Study. In: 19th IEEE Conference on Engineering of Computer Based Systems (ECBS). pp. 210 –221 (April 2012)
38. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Worst-case response-time analysis for mixed messages with offsets in controller area network. In: 17th IEEE Conference on Emerging Technologies and Factory Automation (ETFA) (September 2012)
39. Mubeen, S., Mäki-Turja, J., Sjödin, M., Carlson, J.: Analyzable modeling of legacy communication in component-based distributed embedded systems. In: 37th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). pp. 229–238 (September 2011)
40. Nolin, M., Mäki-Turja, J., Hänninen, K.: Achieving Industrial Strength Timing Predictions of Embedded System Behavior. In: ESA. pp. 173–178 (2008)
41. P. Berggren: Autonomous Cruise Control for Chalmers Vehicle Simulator. Master's thesis, Dept. of Signals and Systems, Chalmers University of Technology (2008)
42. Palencia, J., Harbour, M.G.: Schedulability Analysis for Tasks with Static and Dynamic Offsets. IEEE International Symposium on Real-Time Systems p. 26 (1998)
43. Rajeev, A.C., Mohalik, S., Dixit, M.G., Chokshi, D.B., Ramesh, S.: Schedulability and end-to-end latency in distributed ecu networks: formal modeling and precise estimation. In: EMSOFT, 2010. pp. 129–138. ACM

44. Schmidt, D., Kuhns, F.: An overview of the Real-Time CORBA specification. Computer 33(6), 56 –63 (June 2000)
45. Sha, L., Abdelzaher, T., rzén, K.E.A., Cervin, A., Baker, T.P., Burns, A., Buttazzo, G., Caccamo, M., Lehoczky, J.P., Mok, A.K.: Real Time Scheduling Theory: A Historical Perspective. Real-Time Systems 28(2/3), 101–155 (2004)
46. Stappert, F., Jonsson, J., Mottok, J., Johansson, R.: A Design Framework for End-To-End Timing Constrained Automotive Applications. In: ERTS, 2010
47. Tindell, K.W.: Using offset information to analyse static priority preemptively scheduled task sets. Tech. Rep. YCS 182, University of York (1992)
48. Tindell, K., Clark, J.: Holistic schedulability analysis for distributed hard real-time systems. Microprocess. Microprogram. 40, 117–134 (April 1994)
49. Tindell, K., Hansson, H., Wellings, A.: Analysing real-time communications: controller area network (CAN). In: Real-Time Systems Symposium, 1994. pp. 259 –263

Saad Mubeen is a PhD student at Mälardalen Real-Time Research Centre (MRTC), Mälardalen University, Sweden. His research focus is on modeling and timing analysis of distributed real-time embedded systems in the automotive domain. Saad received his degree of Licentiate in Computer Science and Engineering from Mälardalen University in January 2012. He received his degree of M.Sc. in Electrical Engineering with specialization in Embedded Systems from Jönköping University (Sweden) in 2009. He has co-authored over 30 research papers in peer-reviewed conferences, workshops, books and journals.

Jukka Mäki-Turja is a senior lecturer and researcher at MRTC. His research interest lies in design and analysis of predictable real-time systems. Jukka received his PhD in computer science from Mälardalen University in 2005 with response time analysis for tasks with offsets as focus. He has co-authored over 75 research papers in peer-reviewed conferences, workshops and journals.

Mikael Sjödin is a professor of real-time system and research director for Embedded Systems at Mälardalen University, Sweden. His current research goal is to find methods that will make embedded-software development cheaper, faster and yield software with higher quality. Concurrently, Mikael is also been pursuing research in analysis of real-time systems, where the goal is to find theoretical models for real-time systems that will allow their timing behavior and memory consumption to be calculated. Mikael received his PhD in computer systems in 2000 from Uppsala University (Sweden). Since then he has been working in both academia and in industry with embedded systems, real-time systems, and embedded communications. Previous affiliations include Newline Information, Melody Interactive Solutions and CC Systems. In 2006 he joined the MRTC faculty as a full professor with speciality in real-time systems and vehicular software-systems. He has co-authored over 200 research papers in peer-reviewed conferences, workshops, books and journals.

Received: June 14, 2012; Accepted: November 12, 2012.

DOI: 10.2298/CSIS120423012T

Design of a Multimodal Hearing System

Bernd Tessendorf[1], Matjaz Debevc[2], Peter Derleth[3], Manuela Feilner[3], Franz Gravenhorst[1], Daniel Roggen[1], Thomas Stiefmeier[1] and Gerhard Tröster[1]

[1] Wearable Computing Lab., ETH Zurich
Gloriastr. 35, 8092 Zurich, Switzerland
{lastname}@ife.ee.ethz.ch
[2] University of Maribor
Smetanova ulica 17, 2000 Maribor, Slovenia
{firstname.lastname}@uni-mb.si
[3] Phonak AG
Laubisrütistrasse 28, 8712 Stäfa, Switzerland
{firstname.lastname}@phonak.ch

Abstract. Hearing instruments (HIs) have become context-aware devices that analyze the acoustic environment in order to automatically adapt sound processing to the user's current hearing wish. However, in the same acoustic environment an HI user can have different hearing wishes requiring different behaviors from the hearing instrument. In these cases, the audio signal alone contains too little contextual information to determine the user's hearing wish. Additional modalities to sound can provide the missing information to improve the adaption. In this work, we review additional modalities to sound in HIs and present a prototype of a newly developed wireless multimodal hearing system. The platform takes into account additional sensor modalities such as the user's body movement and location. We characterize the system regarding runtime, latency and reliability of the wireless connection, and point out possibilities arising from the novel approach.

Keywords: multimodal hearing instrument, assistive technology

1. Introduction

Recent studies show that hearing impairment is increasingly affecting people worldwide [18, 27]. The World Health Organization estimates that in 2005 the number of people in the world with hearing impairments was 278 million, or about 4.3% of the world's population [38]. Permanent hearing loss is a leading global health care burden, with 1 in 10 people affected to a mild or significant degree [7].

The demographic change in the European Union (EU) [30] leads to a strong increase in the number of hearing impaired people. At the age of 40 years, the auditory perception begins to deteriorate and beyond 60 more than 50% of the people perceive deterioration of their hearing ability. A report prepared by the Action on Hearing Loss estimates that in 2005 more than 81.5 million

adults in the EU had hearing problems and that this number will increase to 90.6 million by 2015. This figure indicates that more than 14% of adults in Europe will experience hearing problems [26]. Consequently, new technologies to assist people with hearing impairment have emerged. They include: Digital hearing instruments (HIs) [1], cochlear implants [8], implantable hearing devices [31], and more advanced assistive technologies such as the frequency modulation (FM) system [11].

Context-aware HIs automatically adapt to the estimated user's current hearing wish by switching hearing programs, e.g. switching HI programs in quiet or noisy environments, in face-to-face conversation, traffic, or music [2, 6]. By obviating the need for manual selection, these HIs avoid drawing the user's and other's attention to the hearing deficit. This is especially useful in situations where constant change of hearing programs is necessary. Currently, automatic adaption is based on computational auditory scene analysis (CASA [36]), which analyses the acoustic environment for music, conversations, or relative silence. However, in the same acoustic environment an HI user can have different hearing wishes that require different behaviors from the HI. We refer to this as the *ambiguity problem* [32]. In such scenarios, the audio signal alone contains too little contextual information to determine the user's hearing wish. This especially applies for for complex hearing situations with multiple sound sources or different possible activities or movements of the HI user. Therefore, there is a strong need to have additional sources of information available for hearing impaired people to support them in these complex scenarios.

Paper Scope and Contributions In collaboration with an HI manufacturer, HI acousticians and HI users, we developed a wireless multimodal hearing system. To improve the HI adaption in acoustically ambiguous situations, we consider sensor modalities in addition to sound analysis. In our approach we do not need to equip the user's environment with additional hardware. Instead, we developed a miniaturized wireless head movement sensor attachable to commercial HIs. The user's head movement data and the sound feature data from the HI are transferred wirelessly to a smartphone. In a future generation of HIs the accelerometer will be integrated into the HI obviates the need for any additional devices. A dedicated smartphone application fuses the information together with sensor information from the smartphone itself (e.g. GPS or phone acceleration) to derive an improved estimation of the user's hearing wish. We characterize the system regarding runtime, latency and reliability of the wireless connection, and point out possibilities arising from the novel technology.

2. State of the Art in Supporting Hearing Impaired and Review of Additional Modalities to Sound

2.1. Hearing Instrument Technology

Most HIs use digital signal processing to process sounds from the acoustic environment and can be fitted to suit the HI user's individual hearing impairment

for different hearing wishes. The most common types of HIs on the market are behind-the-ear (BTE), in-the-ear (ITE), in-the-canal (ITC) and completely-in-the-canal (CIC). Frequency modulation (FM) systems are used to transmit distant sound directly to an HI user. An approach to overcome difficulties on the phone is to use magnetic induction with a dedicated coil (telecoil, T-coil), which allows different sound sources to be directly and wirelessly connected to the HI regardless of background noise. Figure 1 depicts the components of a BTE HI. Current systems include a digital signal processor, two microphones to enable directivity and conversion of sound, a miniature loudspeaker (receiver) a telecoil, and a high-capacity battery. The sound is conveyed acoustically via a tube to a custom ear mold (omitted in Figure 1). A HI performs the audio processing function of the HI encompassing audio pickup, processing, amplification and playback. The HIs at the user's ears can communicate with each other to stream sound and configuration data. They can also integrate a variety of accessories such as remote controls, Bluetooth or FM devices to form wireless networks, so called hearing instrument body area networks (HI-BANs) [4]. This motivates and supports our investigation of additional sensor modalities for HIs that may eventually be included within the HI itself, or within the hearing system of which the HI is one component. The automatic hearing program selection estimates the user's hearing wish based on the acoustic environment of the given situation and adjusts the sound parameters of the HI from among a set of hearing programs [16]. The classification is based on spectral and temporal features extracted from the audio signal [6] with regard to the audiometry data of the hearing impaired [17]. The hearing programs are optimized for different hearing wishes and selectively use advanced signal processing such as adaptive noise canceling, directivity ("beam forming") or multiband compression. Most current high-end HIs distinguish between four hearing programs: natural hearing (*Clean Speech*), speech intelligibility in noisy environments (*Speech in Noise*), comfort in noisy environments (*Noise*), and pleasure listening for a source with high dynamics (*Music*). Each hearing program represents a trade-off, e.g. between speech intelligibility and naturalness of sound. The automatic selection of hearing programs in HIs according to the user's current acoustic environment allows the hearing impaired to use the device with little or no manual interactions, such as program change. This also avoids drawing attention to the user's hearing impairment. Users consider automatic adaption mechanisms for changing the hearing programs as beneficial [6].

2.2. The Acoustic Ambiguity Problem

State-of-the art HIs which implement the automatic program selection based on auditory information only show intrinsic limitations. They select the most suitable hearing program according to the user's acoustic environment based on computational auditory scene analysis (CASA) [2, 6, 10, 36]. This approach performs well as long as the acoustic environment and hearing wish are directly correlated, e.g. when listening to direct speech in quiet environments. This assumption does not hold in all cases and leads to a limitation we call *Acoustic*

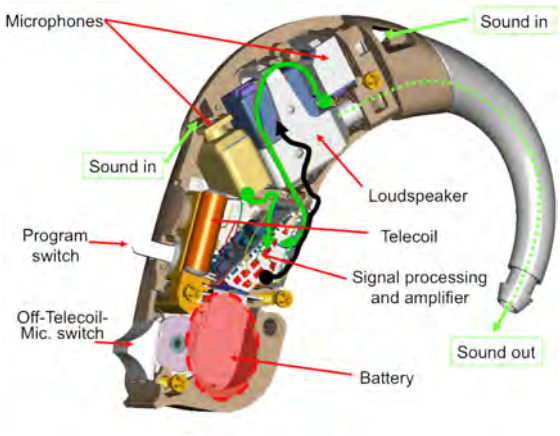


Fig. 1. Components of a behind-the-ear (BTE) HI [25].

Ambiguity Problem [32]. Specifically, in the same acoustic environment a user can have different hearing wishes that require different hearing programs to be active. A sound-based processing cannot distinguish between these different hearing wishes. For example, when there is a person reading a newspaper in a busy train, the HI senses speech in noise. Solely based on this acoustic information, it is not clear whether the HI should optimize the sound processing for speech intelligibility or the user desires a hearing program that provides comfort in noise. Usually, HIs favor to optimize for speech, as social interactions, conversations in particular, are important for HI users. Unfortunately, the hearing impaired person's hearing wish in this case is not to listen to the passengers next to them. However, in a similar situation the passenger could actually favor to participate in a conversation. The HI detects the same acoustic environment and, thus, cannot select a suitable hearing program in both of the cases. Therefore, it is important to not only analyze the acoustic environment but to also assess the relevance of auditory objects [28]. The challenge here is not the effectiveness of the dedicated hearing programs but rather how to automatically adapt the hearing program to the user's specific hearing wish. Other typical situations in which state of the art HI program selection algorithms tend to fail include listening to music from the car radio while driving [13], participating in street traffic as a pedestrian [35], conversing in a cafe with background music [13], and watching TV [35].

2.3. Sensor Modalities Additional to Sound in Hearing Instruments

We can extract contextual information to support automatic hearing program selection using the following approaches:

Head Movements and Mode of Locomotion Head movements carry nonverbal cues during conversations [15]. Hadar et al. [12] found a relation of timing, tempo and synchrony in the listeners' head movements as responses to conversational functions. In our previous work [32] we confirmed head movements and head acceleration in particular, to be a relevant additional sensor modality to recognize the HI user's current hearing wish. We compared accuracies for hearing wish recognition between different on-body sensor positions in an office scenario and found the highest accuracy for the head position [32]. Besides the characterization of conversations, movement patterns at the head can as well be used to recognize head gestures or the user's mode of locomotion. Each of this additionally unveiled contextual information that supports the automatic hearing program selection. Atallah et al. demonstrate a triaxial accelerometer in their study that was placed inside an HI-shaped housing and was worn behind the ear to perform gait analysis [3]. Different activities such as reading, walking, lying down, walking slowly, and running fast could be detected.

Smartphones Smartphones could be used as user interfaces for hearing program selection, for configuration of manual contextual information, for user feedback to improve and personalize automatic program selection or to use the wirelessly connections, and most important as a rich sources of sensors. In a survey among 80 HI users we investigated the availability of smartphones and the users' acceptance for using them to improve HIs [35]. A share of 28% of the respondents always carries their smartphones, another 24% most of the time and we found a clear trend that younger age groups carry the smartphone more often than elderly. About 64% don't have concerns to leverage their phones for the HI, 24% are not sure and demand more information to decide, and the remaining part doesn't want the phone to communicate with the HI. According to this survey smartphones represent promising devices to enhance HIs.

User Location In our previous work [33] we investigated the potential of the user's location to improve automatic hearing program selection. It was evident, that the combination of the user's location with the mode of locomotion reveals significant correlations with the user's current hearing wish. A smartphone can be used to capture the user location via GPS or via wireless network fingerprints. Through the smartphone's connectivity, the internet can be used to associate the raw location to currently ongoing events which may also impact the user's hearing wish. Through these location-aware services, the automatic switching algorithm can consider if the user is listening to an open air concert or he is just walking in a park.

Tagging Tagging refers to putting dedicated beacons to objects. In a study by Hart et al. an attentive HI based on an eye-tracking device and infrared tags was proposed [14]. Wearers can "switch on" selected sound sources such as a person, television or radio by looking at them. The sound source needs to be

augmented with a tag device that catches the attention of the HI user. This way, only the communication coming from the sound sources, which are looked at, are heard.

In their study Choudhury and Pentland propose body-worn IR transmitters that are used to measure face-to-face interactions between people with the goal to model human networks [9]. The success of IR detection depends on the line-of-sight between the transmitter-receiver pair and all partners involved in the interaction needed to wear a dedicated device.

Auditory Selective Attention Capturing the user's auditory selective attention helps to recognise a person's current hearing wish. Research in the field of electrophysiology focuses on mechanisms of auditory selective attention inside the brain [29]. Under investigation are event-related brain potentials using electroencephalography (EEG). In a heart rate analysis, done by Molen et al. the influence of auditory selection on the heart rate was investigated [23]. However, the proposed methods are not robust enough yet to distinguish between hearing wishes in mobile real-life settings.

None of the mentioned approaches for additional sensor modalities have managed to reach integration into off-the-shelf HIs yet because either the performance is too poor or support for deployment in mobile settings is missing. Based on the review above we consider head movements and user location as promising modalities to be integrated into a multimodal hearing system.

2.4. Actuator Modalities Additional to Sound in Hearing Instruments

Besides sensor modalities, research is ongoing considering actuator modalities to enhance HIs. The region behind the ears at the mastoid bone is one of the most sensitive head regions for vibrotactile stimulation [24]. In previous studies we investigated bilateral vibrotactile feedback, integrated into HIs for localisation [34]. An advantage of using vibrotactile feedback is that there is no interference with the sound from the acoustic environment. Moreover, tactile reaction time can be faster than auditory feedback [22].

In the study from Borg et al. a pair of glasses were enhanced with 4 vibrators and 3 microphones [5]. Sound source angles were located through the integrated microphones and translated to vibration patterns for the visually or hearing impaired wearer of the glasses. The approach did not focus on integration into HIs, instead the user needs to wear the proposed enhanced goggles. The results show an average share of correctly detected sound source angles of about 80% in a sound-treated room.

Weisenberger et al. present a vibrating device to be placed inside the ear mold to transduce sound into vibration [37] with a vibration frequency of 80 Hz for low frequency acoustic signals and 300 Hz for high frequency acoustic signals. Subjects have been tested in three tasks: sound localization, sound identification, and syllable rhythm and stress. Overall, the ear mold vibrator system

showed promising results as an actuator modality additional to sound, especially for aiding sound localization.

3. Multimodal Approach

By complementing sound with contextual information from additional information channels we provide the means to improve the automatic HI adaption in acoustically ambiguous situations. Based on our review of additional modalities to sound in the previous section we introduce a newly developed wireless multimodal hearing system, which takes into account the user's head movements and location. Previous studies have shown that additional modalities, head movements in particular, improve automatic hearing program selection [32,33]. Furthermore, the integration of the head movement sensor allows the users to control their HIs using head gestures.

3.1. Architecture of the Multimodal Hearing System

Figure 2 depicts an overview of the architecture of the multimodal hearing system and its communication. Commercial HIs are extended with a miniaturized triaxial acceleration sensor and communicate sound and acceleration data to the user's smartphone. The user wears a commercially available vendor-specific relay (shown here: Phonak iCube) around his neck to establish a wireless communication between the HI and the smartphone. The commercially available relay translates the bidirectional communication between the proprietary wireless protocols used by the HIs to a commonly used protocol, e.g. Bluetooth. The relay can stream sound from a phone or TV or play music from portable devices.

The protocols used for the system communication are denoted in brackets. The smartphone invokes an updated hearing program based on the analysis of the multimodal information. The system architecture is:

- *opportunistic*, i.e. the system falls back to a working stand-alone HI if the user's smartphone is currently not available, and
- leveraging a *smartphone* to collect and process sensor data and user interaction with the smartphone,
- *modular and scalable*, it is not limited to the selected set of modalities but can be extended to further modalities using standard protocols like WiFi, ANT+ or Bluetooth,
- *backward-compatible*, i.e. older HIs that support a relay or remote control can be upgraded with this technology.

Detailed descriptions of the architecture's building blocks are given in the following sections.

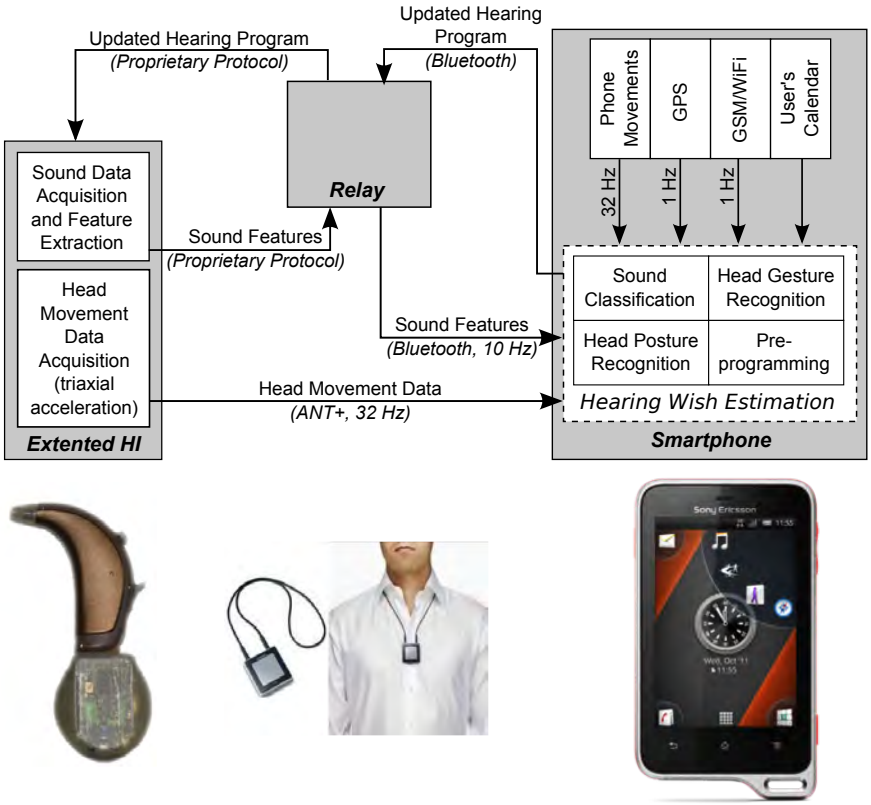


Fig. 2. Architecture of the developed prototype of a wireless multimodal hearing system.

3.2. System Communication

The communication paths between the system components and the corresponding communication protocols are shown in Figure 2. The smartphone communicates via Bluetooth with a vendor-specific relay (in our case a Phonak iCube as shown in the middle of Figure 2). The relay communicates over a proprietary wireless protocol with the HIs. Head movement data is sent to the user's smartphone. The acceleration sensor wirelessly communicates via the ANT+ protocol with the smartphone. The ANT+ protocol is a low power protocol designed for reliable transfer of data between sensors and display devices such as watches, heart rate monitors and bike computers. It ensures interoperability to guarantee seamless digital wireless communication in the 2.4 GHz license-free band. The transmitted data can be secured with a private network key. The adjustable sensor sampling and transmission rates are 32 Hz, which is sufficient for most activity recognition tasks [21]. The two HIs (HI model Phonak BTE Ambra 2012) worn at both ears can communicate with each other to syn-

chronize manual hearing program switches and stream sound data. They wirelessly receive hearing program change requests invoked from the smartphone via the relay. In turn, the HIs send sound features, e.g. the sound level, to the smartphone. The smartphone acts like an automatic remote control. Our opportunistic approach ensures an automatic fallback to the original functionality of the HI in case the smartphone or the other additional system components are not available.

3.3. Extension of Hearing Instruments with a Head Movement Sensor

To produce the housing of the modular HI extension shown in Figure 3 we used a 3D CAD rapid prototyping method based on an acrylic photopolymer material. The head movement sensor has the dimensions of 22.6 mm $\times$ 21.6 mm $\times$ 10.3 mm and weighs 5.1 g (a typical HI weighs 4.7 g) and is attachable to commercial HIs. HIs to be used with the head movement sensor need to feature a slide mechanism at the lower end of the HI housing to mount the device. Most of the commercial HIs have this slide mechanism available to attach for example accessory FM receivers (replacements for the battery compartments for different types of HI are offered, which have the additional slide mechanism). The newly developed head movement sensor is based on the BodyANT platform [19]. It integrates a triaxial acceleration sensor (Bosch SMB380) and is powered by a 140 mAh CR1632 coin cell battery. The battery is placed in a battery compartment and can be replaced by using a coin to turn and open the battery cover. Acceleration can be measured with a bandwidth of up to 1.5 kHz in ranges of $\pm$2 g/$\pm$4 g/$\pm$8 g corresponding to a resolution of 4.0 mg/7.8 mg/15.6 mg. A stable clock cycle is provided using a 16 MHz crystal as clock source for both the radio transceiver and microprocessor. When active, the microprocessor periodically reads sensor values and sends messages to the radio transceiver according to the ANT message protocol. The transceiver continuously broadcasts the messages at a predefined message rate. If not activated, the microprocessor and radio transceiver are kept in power save mode. As mentioned before, the sensor allows the system to capture the user's head movements which is beneficial for improving HIs [32]. Due to the progress in the miniaturization of microelectromechanical systems (MEMS) and the reduction of power consumption of MEMS this technology manages to meet appropriate comfort requirements demanded by HI users [35].

To capture the HI user's head movements we opted for a modular solution, which is attachable to most of the state-of-the-art commercially available HIs. The design decision for an additional piece of hardware compares to integration of an acceleration sensor into the HI itself as follows:

- *Availability*: The modular solution is available now for all compatible state-of-the-art HI models; integration into the HI itself takes at least the time of an HI product development cycle for each single HI model we want to support.
- *Production costs for low volumes*: Our rapid-prototyping solution is cost-effective compared to the complete production of a next generation HI.

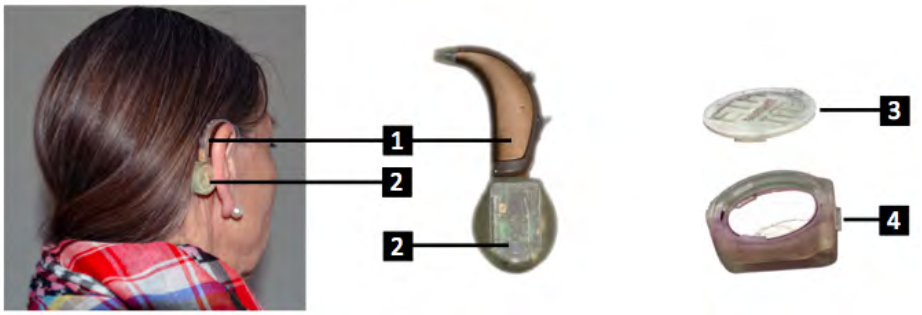


Fig. 3. The wireless multimodal hearing system comprises conventional commercial HIs (1) and the attachable head movement sensor (2). On the right the battery cover (3) and the slide mechanism (4) that sticks out to be attached to the HI, are shown.

- *Research platform*: Large scale real-life evaluations with HI users are needed before HI manufacturers decide to invest in this technology. These kinds of studies are feasible with our proposed platform. The modular approach to the multimodal HI extension allows to evaluate the benefits and limitations of multimodal HIs before a later integration of the additional modalities into the HI housing itself.
- *Backward compatibility*: The head movement sensor can be attached to any older HI that features a slide at the bottom, which is the case for most of the HI devices on the market. It represents a way to upgrade previous HI product generations.
- *System interoperability*: Only the communication with the HI relay is vendor dependent and needs to be adapted for different brands of HIs, the remaining system is vendor independent.

The main advantage of integration of the sensors into HIs over the modular solution are shared hardware resources, in particular the micro controller and transceiver for wireless communication. This way a saving in power consumption and form factor could be realized. Thus, both approaches have advantages and represent parallel solutions.

3.4. Smartphone Application

Smartphones are becoming the central computer and communication device in people's lives [20]. We leverage a smartphone as a component of the multimodal hearing system for the following reasons:

- *Processing power*: With an uprising trend modern smartphones offer processing resources up to 2000 MIPS, e.g. to execute complex context recognition algorithms.

- *Availability*: In previous work we identified smartphones to be available and accepted by their users [35].
- *Sensors*: Smartphones provide a rich source of sensor information such as an accelerometer, digital compass, gyroscope, GPS, microphone, WiFi, Bluetooth, ANT+, and camera.
- *Connectivity and scalability*: Smartphones provide internet access for cloud connectivity, access to the user's calendar, and support standard wireless protocols to extend the system with additional sensors in a modular way.
- *Extensibility*: Smartphones are programmable and additional applications can be developed, leveraging crowd sourcing and community driven software development.
- *User interface*: The smartphone can provide the user with a GUI to change more complex settings than possible with the buttons of the HI.

The newly developed smartphone application runs on any Android based phones that support the ANT+ protocol (we used a Sony Ericsson Xperia active smartphone). We opted for an Android-based software approach because it provides open source software development tools. The smartphone application performs the following tasks:

- receive and process sound features and acceleration data,
- provide a visual real-time data presentation of the sensor data,
- process the smartphone's local sensors,
- read the user's calendar and activate HI settings based on calendar entry, calendar entries that start with the special tag *HI:* are parsed by the smartphone application.
- use the user's location to activate room specific HI settings, e.g. reverberation characteristics of a concert hall; the application is prepared to download location-specific HI settings from a database from the cloud. This database can be populated by HI users to form a virtual HI community to share HI data, e.g. users can label their hearing wish in a specific place,
- perform classification of the sensor data,
- allow for data annotation, which is useful for HI developers and for HI end users also to train the HI using machine learning algorithms to let the HI automatically adapt to new context situations,
- enables to remotely log into the smartphone for debugging (with considerations for data anonymization and usage according to privacy laws and user agreement).

Figure 4 depicts screenshots of the smartphone application showing a data visualisation of sound features (hearing program class probabilities calculated within the HI, root mean square (RMS) sound level) and head acceleration, and a GUI that allows the user to program the HI using the smartphone's calendar. In this example the user programmed the hearing program "Comfort in Noise" to become active when traveling with public transport to his workplace, because he usually reads a newspaper and does not want the HI to optimize to the conversations around him. The smartphone application parses calendar entries

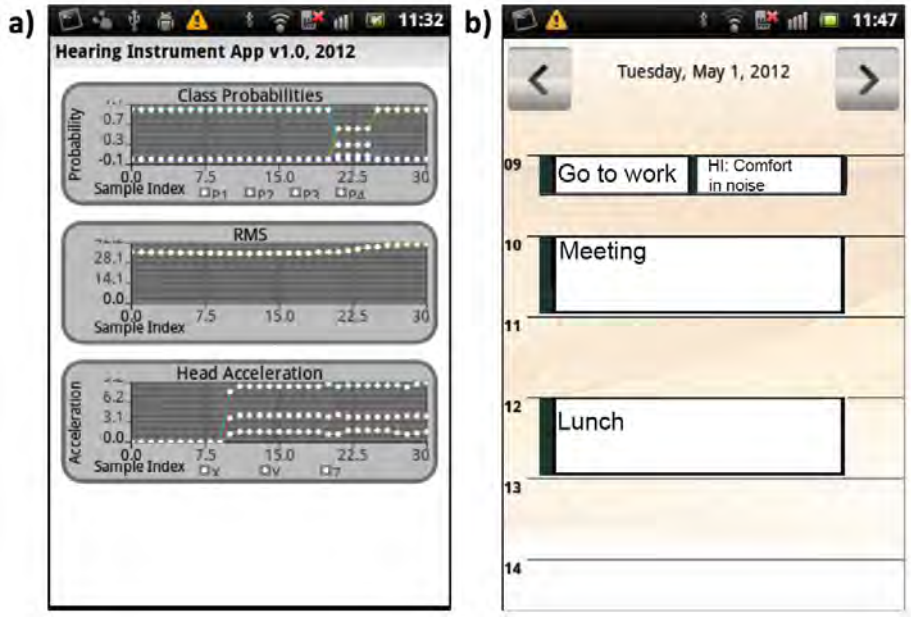


Fig. 4. Screenshots of the smartphone application: (a) Data visualisation of sound features and head acceleration, and (b) a GUI to program the HI using the built-in calendar.

that start with the special tag "*HI:*" and sets the hearing program accordingly at the start time of the entry.

4. System Characterization

4.1. Power Consumption

Battery lifetime is a critical factor for HI users [35]. The battery runtime for the head movement sensor with a 140 mAh CR1632 coin cell battery is more than 17 hours when the sensor sampling and transmission rates are 32 Hz. It increases to more than 4 days when the rate is reduced to 16 Hz, which is still sufficient for many activity recognition tasks [21]. The battery lifetime does not directly correlate with the power consumption due to the nonlinear discharge curve of the battery. Figure 5 shows the power consumption of the head movement sensor for different sampling rates. The ANT+ transmission rate was the same as the sensor sampling rate for the measurements. The runtime for the smartphone application is more than 16 hours with a 1200 mAh LiIon battery (3.7 V) when no other additional applications are being executed. Figure 6 shows the share of different components for the power consumption of the smartphone. We obtained the values by measuring the current from the

battery when having the different components individually activated. The runtime is sufficient for everyday use when it is recharged overnight. The battery lifetime of the relay we used is up to a 10 hours.

4.2. Packet Loss

We measured the packet loss occurring during the transmission from the sensor to the smartphone for the user wearing the smartphone on different locations on the body: left and right front trouser pocket, left and right back trouser pocket, and left and right upper arm, attached with a strap that was shipped with the smartphone. The HI with the head movement sensor was worn at the left ear. For each phone location the user performed activities of daily living including sitting, standing, and walking. We calculate the packet loss as the rate of data packets that were not received at the smartphone using Equation 1:

$$Packet\ Loss := \frac{\#Sent\ packets - \#Received\ packets}{\#Sent\ packets} \tag{1}$$

Table 1 shows the measured packet loss values. Packet loss is low for all smartphone locations and renders the wireless communication suitable for application in multimodal hearing systems. The largest packet loss value was on the back right trouser pocket, where the user's body is in the line of sight of the transmission path, this way damping the signal. We did not observe any packet loss from the smartphone to the HI.

4.3. Latency

Latency refers to the time between the onset of a head movement and the time the system reacts to it. The total latency is below 100 ms and is comprised as follows: The communication delay between head movement sensor and smartphone is below 6 ms. Acceleration data is usually evaluated using block processing with a sliding window. The main influence on the latency is the window

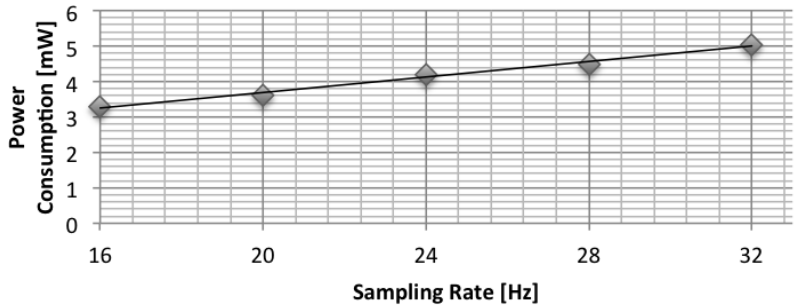


Fig. 5. Power consumption of the head movement sensor for different sampling rates.

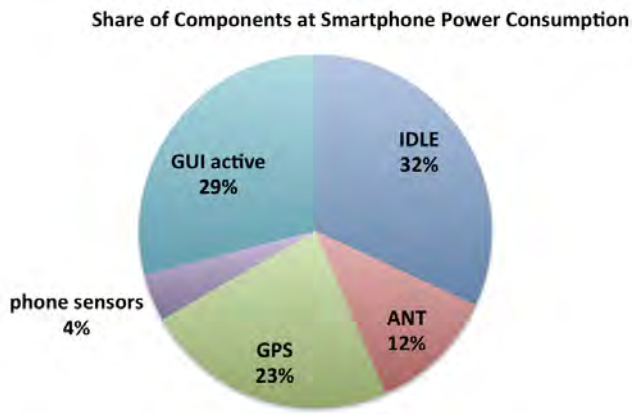


Fig. 6. Share of different components for the power consumption of the smartphone.

Table 1. Packet loss for the system communication between head movement sensor and the smartphone on different locations on the body.

Smartphone Location	trouser front pocket		trouser back pocket		upper arm	
	left	right	left	right	left	right
Packet Loss [%]	0.04	0.16	0.71	1.07	0.10	0.07

size used and is in the range of the duration of the head movement to be detected. The roundtrip communication time between the smartphone and the HI via the relay is 60 ms to 90 ms. Based on the above results we conclude the wireless communication of the system is fast enough to transfer sound features and head movement data and set HI programs.

4.4. Potential of the Multimodal Hearing System

We designed the multimodal hearing system as a flexible platform to pave the way for use in a broad spectrum of applications. Its pervasive usability, small size, flexibility and possibility for long-term deployment allows us to bring the efforts from various research domains into daily use. Besides passive HI control through recognition of the user's context and active HI control, e.g. through user head gestures, we see further applications of the platform at least in the following research domains:

- activity recognition and pervasive computing,
- human computer interaction (HCI),
- computational social science,
- long-term behavior monitoring, and
- self-adapting and -learning systems.

4.5. Limitations

A crucial aspect to make the system usable and reliable is to ensure a low complexity. The prototype allows for both automatic and manual control of the HI. It has to be mentioned that additional manual interaction with the smart phone contributes to high user interaction costs. For the automatic hearing program selection, however, the context is derived from the user's movement and the user does not have to change his behavior or learn to use the system. In a future generation of HIs the accelerometer will be integrated into the HI and there will be no need to carry additional devices such as the mobile phone to benefit from improved automatic hearing program selection. Additional functions such as programming the HIs using the phone's calendar, or using head gestures require more interaction and training. of the user. The amount of required training will depend on the technical background of the user and the actual user interface for these features. This has to be studied and optimized in additional experiments.

The head movement sensor is implemented as an additional device attached beneath a HI, it is not yet integrated inside the HI itself. Therefore we face some additional limitations:

- Users of the prototype system need to carry the relay and phone as additional devices with additional weight and battery maintenance to benefit from any of the new functionalities.
- Our presented setup requires an Android-operated smart phone which features the ANT+ protocol. Up to now, there are only a very limited number of phones which support ANT+. We opted for ANT+ since this technology is consuming less power than conventional bluetooth transmission and might be widely established in the near future. However, the setup can easily be adapted to work with bluetooth to ensure compatibility with older phones. Alternatively, the Bluetooth low energy protocol could be used. In any case the used wireless protocol should be standardized across HI and smartphone manufacturers, and could be integrated into the HI's relay.

However, these limitations will become obsolete for future generations of HIs that integrate the accelerometer. When the accelerometer is integrated into the HI, we will face a reduction in the HI's battery lifetime. However, the additional power consumption cannot be quantified as long as the additional functionalities are not finally integrated into the HI. Possible optimizations concerning battery lifetime strongly depend on the actual implementation and applied power management techniques of the final integrated device. We expect the impact to be low due to the availability of low-power MEMS accelerometers (e.g. 250 uA for the ST LIS331H accelerometer).

5. Conclusion and Future Work

We presented a newly developed wireless multimodal hearing system. It represents an enabling technology, which raises new possibilities for HI users, HI acousticians and HI manufacturers:

- to improve automatic hearing program selection in acoustically ambiguous situations using additional sensor modalities,
- to implement and investigate the benefit of a gesture controlled HI,
- to introduce location aware support,
- to let the HI user schedule his daily routines and corresponding HI programs,
- to allow HI manufacturers remote debugging capabilities in the field to improve the product (with considerations for data anonymization and usage according to privacy laws and user agreement)
- to support the HI acoustician with fitting the HI to the user by providing multimodal contextual information from real-life situations, in which the user's appreciate modified HI sound settings

With a day of battery lifetime, reliable wireless connection and sufficiently small latency (below 100 ms), we found the system to be viable both as a research platform and as a working prototype for a potential product in the HI market. The head movement sensor can be used to upgrade previous HI generations and enables evaluations towards integration of sensors into HI itself.

In future work we plan to conduct long term real-life studies with HI users. Besides assessing the user acceptance, we want to confirm the benefit of multimodal hearing systems, already demonstrated in laboratory settings [32], for real-life situations. We further plan to assess the benefit of a head gesture controlled HI, particularly for elderly people.

Acknowledgments. This work was part funded by CTI project 10698.1 PFLS-LS. We especially thank Nadim El Guindi and Stephan Koch for valuable discussions and support with the relay framework.

References

1. Abrams, H.: Digital hearing aids. Ear and Hearing 30(3), 385 (2009)
2. Allegro, S., Büchler, M., Launer, S.: Automatic sound classification inspired by auditory scene analysis. In: Consistent and Reliable Acoustic Cues for Sound Analysis (CRAC), one-day workshop, Aalborg, Denmark, Sunday September 2nd 2001 (directly before Eurospeech 2001). Citeseer (2001)
3. Atallah, L., Aziz, O., Lo, B., Yang, G.Z.: Detecting walking gait impairment with an ear-worn sensor. International Workshop on Wearable and Implantable Body Sensor Networks 0, 175–180 (2009)
4. Biggins, A.: Benefits of wireless technology. Hearing Review (11 2009)
5. Borg, E., Ronnberg, J., Neovius, L., Lie, T.: Vibratory-coded directional analysis: Evaluation of a three-microphone/four-vibrator DSP system. J. of rehabilitation research and development 38(2) (2001)
6. Buchler, M., Allegro, S., Launer, S., Dillier, N.: Sound Classification in Hearing Aids Inspired by Auditory Scene Analysis. EURASIP Journal on Applied Signal Processing 18, 2991–3002 (2005)
7. Cavender, A., Ladner, R.: Hearing impairments. Web Accessibility pp. 25–35 (2008)
8. Chapman, R.: Cochlear implants. Ear and Hearing 29(3), 477 (2008)

9. Choudhury, T., Pentland, A.: Sensing and modeling human networks using the sociometer. In: ISWC. p. 216. IEEE Computer Society, Washington, DC, USA (2003)
10. Eronen, A., Peltonen, V., Tuomi, J., Klapuri, A., Fagerlund, S., Sorsa, T., Lorho, G., Huopaniemi, J.: Audio-based context recognition. IEEE Transactions on speech and audio processing 14(1), 321 (2006)
11. Fitzpatrick, E., Séguin, C., Schramm, D., Armstrong, S., Chénier, J.: The benefits of remote microphone technology for adults with cochlear implants. Ear and hearing 30(5), 590 (2009)
12. Hadar, U., Steiner, T.J., Clifford Rose, F.: Head movement during listening turns in conversation. Journal of Nonverbal Behavior 9(4), 214–228 (12 1985)
13. Hamacher, V.: Signal processing in high-end hearing aids: State of the art, challenges, and future trends. EURASIP Journal on Applied Signal Processing 18(2005), 2915–2929 (2005)
14. Hart, J., Onceanu, D., Sohn, C., Wightman, D., Vertegaal, R.: The attentive hearing aid: Eye selection of auditory sources for hearing impaired users. Human-Computer Interaction –INTERACT (2009)
15. Heylen, D.: Challenges Ahead. Head Movements and other social acts in conversation. In: AISB 2005, Social Presence Cues Symposium (2005)
16. Keidser, G.: Many factors are involved in optimizing environmentally adaptive hearing aids. The Hearing Journal 62(1), 26 (2009)
17. Keidser, G., Yeend, I., O'Brien, A., Hartley, L.: Using in-situ audiometry more effectively: How low-frequency leakage can effect prescribed gain and perception. In: Hearing Review. vol. 18, pp. 12–16 (2011)
18. Kochkin, S.: MarkeTrak VIII: 25-year trends in the hearing health market. Hearing Review 16(10) (2009)
19. Kusserow, M., Amft, O., Tröster, G.: Bodyant: Miniature wireless sensors for naturalistic monitoring of daily activity. In: BodyNets (2009)
20. Lane, N., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., Campbell, A.: A survey of mobile phone sensing. Communications Magazine, IEEE 48(9), 140–150 (2010)
21. Maurer, U., Smailagic, A., Siewiorek, D., Deisher, M.: Activity recognition and monitoring using multiple sensors on different body positions. In: Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. IEEE (2006)
22. Mohebbi, R., Gray, R., Tan, H.: Driver reaction time to tactile and auditory rear-end collision warnings while talking on a cell phone. Human Factors: The Journal of the Human Factors and Ergonomics Society 51(1), 102–110 (2009)
23. Molen, M., Somsen, R., Jennings, J.: Does the heart know what the ears hear? A heart rate analysis of auditory selective attention. Psychophysiology (1996)
24. Myles, K.: Guidelines for Head Tactile Communication. Tech. rep., Army Research Lab Aberdeen Proving Ground Md Human Research And Engineering Directorate (2010)
25. Naylor, G.: Modern hearing aids and future development trends
26. Rohdenburg, T., Huber, R., van Hengel, P., Bitzer, J., Appell, J.: Hearing aid technology and multi-media devices. ITG Fachtagung für Elektronische Medien 13 (2009)
27. Shargorodsky, J., Curhan, S., Curhan, G., Eavey, R.: Change in Prevalence of Hearing Loss in US Adolescents. JAMA 304(7), 772 (2010)
28. Shinn-Cunningham, B.: I want to party, but my hearing aids won't let me! In: Hearing Journal. vol. 62, pp. 10–13 (2009)
29. Shinn-Cunningham, B., Best, V.: Selective attention in normal and impaired hearing. Trends in Amplification 12(4), 283 (2008)
30. Silton, W., Schoenefeld, J.: An aging europe. demographic transition and economic implicatios (2010)

31. Snik, A.: Implantable hearing devices for conductive and sensorineural hearing impairment. In: Auditory Prostheses, Springer Handbook of Auditory Research,, vol. 39, pp. 85–108. Springer New York (2012)
32. Tessendorf, B., Bulling, A., Derleth, P., Feilner, M., Roggen, D., Stiefmeier, T., Tröster, G.: Recognition of hearing needs from body and eye movements to improve hearing instruments. In: International Conference on Pervasive Computing (2011)
33. Tessendorf, B., Derleth, P., Feilner, M., Kettner, A., Roggen, D., Stiefmeier, T., Tröster, G.: Identification of relevant multimodal cues to enhance context-aware hearing instruments. In: 6th International Conference on Body Area Networks (Bodynets) (2011)
34. Tessendorf, B., Derleth, P., Feilner, M., Roggen, D., Stiefmeier, T., Tröster, G.: Improving game accessibility with vibrotactile-enhanced hearing instruments. In: 13th International Conference on Computers Helping People with Special Needs (ICCHP), Linz, Austria (2012)
35. Tessendorf, B., Derleth, P., Feilner, M., Roggen, D., Stiefmeier, T., Tröster, G.: Survey on the potential of additional modalities for hearing instruments. In: Proc. of the 38th Annual Convention for Acoustics (DAGA 2012) (2012)
36. Wang, D., Brown, G.: Computational auditory scene analysis: Principles, algorithms, and applications. IEEE Press (2006)
37. Weisenberger, J., Heidbreder, A., Miller, J.: Development and preliminary evaluation of an earmold sound-to-tactile aid for the hearing-impaired. J Rehabil Res Dev 24, 51–66 (1987)
38. WHO: Deafness and hearing impairment, Fact sheet N°300, February 2012, http://www.who.int/mediacentre/factsheets/fs300/en/index.html
39. Wilson, B., Dorman, M.: Cochlear implants: a remarkable past and a brilliant future. Hearing research 242(1-2), 3–21 (2008)

Bernd Tessendorf received the Diploma degree in Electrical Engineering and Information Technology (Dipl.-Ing.) from RWTH Aachen University, Germany, in 2006. In 2007, he received the Diploma degree in Economics (Dipl.-Kfm.) from Fernuniversity Hagen, Germany. In 2008, he joined the Wearable Group at the Electronics Laboratory at ETH Zurich.

Matjaž Debevc received Ph.D. degree in computer science from University of Maribor in 1995. He is currently an Associate Professor in Computer Science at the Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include human-computer interaction, e-learning, user interface design, adaptive user interfaces, internet applications, interactive TV, distance education and applications for disabled people.

Peter Derleth received Ph.D. degree in Physics form University of Oldenburg, Germany, in 1999. Since 2000 he is employed by Phonak AG, Switzerland. Since 2005 he is leading the Algorithm Concepts Group in the Research Department.

Manuela Feilner studied Electrical Engineering at ETH Zurich and received her Ph.D. degree from EPFL (Swiss Federal Institute of Technology Lausanne)

in 2002. Since 2003 she works for Phonak AG, Switzerland. Since 2008 she is working in the group of Advanced Concepts and Technologies.

Franz Gravenhorst received his Diploma (Masters) degree in Electrical Engineering and Information Technology at Karlsruhe Institute of Technology, Germany, in 2010. He worked at the Research and Development department of an automotive company in Detroit, USA. He joined the Wearable Group at ETH to start his PhD in 2010.

Daniel Roggen received PhD degree at the Laboratory of Intelligent Systems of EPFL, Switzerland, in 2005. In his PhD he developed bio-inspired electronic circuits with fault-tolerance, learning, and developmental capabilities. Since 2005 he is Senior Researcher in the Wearable Computing Lab at ETH Zurich.

Thomas Stiefmeier is a senior member of the research staff at the Electronics Laboratory at ETH Zurich. He received his master's degree in electrical engineering from the University of Technology Darmstadt and his PhD degree from ETH Zurich. Thomas Stiefmeier is CEO and a co-founder of Amphiro AG, a start-up company in the emerging field of smart water metering.

Gerhard Tröster studied electrical engineering in Darmstadt and Karlsruhe, Germany, earning his doctorate in 1984 at the Technical University of Darmstadt about the design of analog integrated circuits. During the eight years he spent at Telefunken (Atmel) Heilbronn, he headed various national and international research projects centered on the key components for ISDN and digital mobile phones. Since 1993 he directs the Electronics Laboratory at the ETH. In 1997 he co-founded the spin-off u-blox ag.

Received: April 23, 2012; Accepted: November 23, 2012.

DOI:10.2298/CSIS120517013S

Optimization and Implementation of the Wavelet Based Algorithms for Embedded Biomedical Signal Processing

Radovan Stojanović[1], Saša Knežević[1], Dejan Karadaglić[2], and Goran Devedžić[3]

[1] University of Montenegro, Faculty of Electrical Engineering, Montenegro
stox@ac.me, sasaknezevic@live.com

[2] Glasgow Caledonian University, School of Engineering and Built Environment, UK
Dejan.Karadaglic@gcu.ac.uk

[3] University of Kragujevac, Faculty of Engineering, Serbia
devedzic@kg.ac.rs

Abstract. Existing biomedical wavelet based applications exceed the computational, memory and consumption resources of low-complexity embedded systems. In order to make such systems capable to use wavelet transforms, optimization and implementation techniques are proposed. The Real Time QRS Detector and "De-noising" Filter are developed and implemented in 16-bit fixed point microcontroller achieving 800 Hz sampling rate, occupation of less than 500 bytes of data memory, 99.06% detection accuracy, and 1 mW power consumption. By evaluation of the obtained results it is found that the proposed techniques render negligible degradation in detection accuracy of -0.41% and SNR of -2.8%, behind 2-4 times faster calculation, 2 times less memory usage and 5% energy saving. The same approach can be applied with other signals where the embedded implementation of wavelets can be beneficial.

Keywords: wavelet transform, microcontroller, QRS, denoising.

1. Introduction

The Fourier Transform (FT) is an extremely important and useful tool in signal processing. However since it in its original form treats the global signal in its entirety, it has the drawback that some time-local specific features and peculiarities, especially if they occur rarely, well may be lost in the analysis. This limitation can be partly overcome by the introduction of Short Time Fourier Transform (STFT), which uses a sliding time window of fixed length to localize the analysis in time. Among a number of alternative time–frequency methods, the most promising seems to be the Wavelet Transform (WT) [1]. In contrast to FT, which is restricted to the use of a sinusoid, the WT uses a variety of basic functions, known as wavelets [1]. In its discrete form (DWT),

based on orthogonal wavelet, it is particularly useful in signal compression, detection of local discontinuities, feature extraction, filtering ("de-noising") and other applications [2],[3],[4].

Among others, the DWT has been applied to a wide range of biomedical (BME) signals, including Electrocardiogram (ECG), Electromyogram (EMG), Electroencephalograph (EEG), Photoplethysmograph (PPG), clinical sounds, respiratory patterns, blood pressure trends and DNA sequences [5]. Existing applications perform its calculations off-line using desktop computers or servers with special software or mathematical tools, like MATLAB. The input data are prerecorded in special database such as MIT-BIH, QT, etc, and then later analyzed. Also, data can be imported from memory cards of logger devices, like holters. Such calculations suffer from limited autonomy, bulkiness and obtrusiveness and prevent timely action to the patient.

Recently, a surge in industrial, research and academic interest into telemedicine and home care has been noticed, where low-cost, miniature, telemetry devices overcome the distance barrier between the doctor and patient, e.g. remote vital signs monitors [7], [8]. Such devices are, in fact, Systems on Chip (SoC), consisting of a single Microprocessor/Microcontroller (MC) [9], Programmable Logic Device (PLD) or Application-Specific Integrated Circuit (ASIC). In addition to the sensing, digitalization, data storage, visualization and communication, such chips need to perform real-time signal processing even in time-frequency domain. This is not a trivial task considering the limitations in arithmetic power, memory and power consumption resources.

This paper presents a methodology and techniques to implement WT in low-complexity fixed point embedded architectures, like existing low-cost MCs. The real-time QRS detector and "de-noising" filter are implemented in a 16-bit MC from TI's MSP430 series [6]. For these purposes, the Haar wavelet transform is rewritten from floating point to integer arithmetic. The approach resulted in increased processing speed, minimized memory request and decreased power consumption. The detection accuracy of QRS complexes and signal to noise ratio (SNR) remains on satisfactory level. In addition, the MC is capable to output wavelet and "de-noised" coefficients in the form of analog signal and the RR intervals in the form of digital impulses or in the form of ACSII strings.

The work is organized as following: short introduction on WTs; the proposed optimization techniques; application of WTs in QRS detection and "de-noising" as well as an overview of related work are given in Section 2 and Section 3. Section 4 describes the corresponding hardware and software architectures with associated components and algorithms. The testing procedure and results obtained against qualitative and quantitative criteria are elaborated and discussed in Section 5. The conclusion and references used are enclosed at the end.

2. Related Work

In existing literature, there are several contributions on using ASICs and Field Programmable Gate Arrays (FPGAs), which are a type of PLDs, in wavelet-based processing of biomedical signals, and especially ECG. The paper [10] presents QRS detection algorithm implemented in ASIC with 0.18 μm CMOS technology, consuming 176 μW, under 1.8 V supply voltage. The algorithm is based on the Dyadic Wavelet Transform and Multiscale-product Scheme. The algorithm is evaluated on the MIT-BIH database, achieving a high accuracy, >99%. In work [11] the authors propose a structure of QRS detector, which concludes Wavelet Filter Banks and Multi-scale Products to increase detection performances. The filters with Quadratic Spline Wavelet function are chosen to reduce leakage and dynamic power consumption. The design had been prototyped on an Altera's Cyclone-FPGA and synthesized on 0.18 μm Samsung libraries. The paper [12] proposes the algorithm and hardware architecture for QRS detection system based on Mathematical Morphology and Quadratic Spline Wavelet transform, with implementation in Xilinx Virtex^{TM}-4SX35 FPGA. The detection accuracy for MIT/BIH arrhythmia database records and resource consumption are reported and seems to be very high. To filter ECG signal and to extract QRS signs the authors in [13] employ the Integer Wavelet Transform. Their system includes several components, which are incorporated in a single FPGA chip from Altera Cyclone Series, achieving sufficient accuracy (about 95%), remarkable noise immunity and low cost.

One of the first references to the introduction of Digital Signal Processors (DSP) in real time processing of ECG signal, by using wavelets, is given in [14]. In particular, QRS complexes, P and T waves are distinguished from noise, baseline drift or artefacts by SPROC-1400 DSP running on 50 MHz. Follow the implementations on modern DSPs, like TI TMS320C6713 [15], where ECG signal is processed in real-time by using DWT and Adaptive Weighting Scheme. An increasing emphasis has been placed in recent years on approaches based on highly integrated, low-power, low-cost MCs like PICs (from Microchip) [16] or MSP430s (from TI). However, their algorithms are still based on traditional methods based on cascade of derivative and averaging filters.

Although much faster, the ASICs and PLDs are more expensive, power demanding, bulky and complicated for rapid prototyping, massive production and maintenance. Thus, the MC remains to be an appropriate solution and a variety of biomedical algorithms, including those WT based, need to be adopted for using in this technology.

3. Methodology

3.1. WT and DWT

Analytically, the continuous form of WT for a signal *f(t)* is defined by:

$$W(a,b) = \int_{-\infty}^{\infty} f(t)\,\psi_{a,b}(t)dt\,, \tag{1}$$

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi^*\left(\frac{t-b}{a}\right), \tag{2}$$

where * denotes complex conjugation and $\psi_{a,b}(t)$ is a window function called the daughter wavelet, a is a scale factor and b is a translation factor. Here, $\psi^*(\frac{t-b}{a})$ is a shifted and scaled version of a mother wavelet $\psi(t)$, which is used as a basis for wavelet decomposition. However, the continuous wavelet transform provides certain amount of redundant information.

Discrete form of WT, known as DWT, is sufficient for most practical applications, providing enough information and offering a significant reduction in the computation time. For a discrete function *f*(*n*), it is given by:

$$W(a,b) = C(j,k) = \textstyle\sum_{n\epsilon Z} f(n)\psi_{j,k}\,(n)\,, \tag{3}$$

where $\psi_{j,k}(n)$ presents a discrete wavelet defined as $\psi_{j,k}(n) = 2^{-\frac{j}{2}}\psi(2^{-j}n - k)$. The parameters a, b are defined as $a = 2^j$ and $b = 2^j k$.

In practice, DWT is computed by passing the signal through a Low-Pass (L_d) and a High-Pass (H_d) filters successively, according to the Mallat's decomposition scheme, Fig. 1 [17]. For each decomposition level *i*, $1 \le i \le N$, the L_d and H_d filters are followed by a downsampling operator, ↓2 expressed as $(X\downarrow 2)[n]=X[2n]$, which is in fact the reduction of a sampling rate by 2. $CA_i(n)$ and $CD_i(n)$ are approximate and detailed coefficients for i^{th} decomposition level. The number of coefficients for i^{th} decomposition level is equal to l_i=length($CA_i(n)$)=length($CD_i(n)$)=$length(X(n))/2^i$. The reconstruction consists of upsampling by ↑2 and filtering by filters L_r and H_r. The L_d, H_d, L_r, H_r coefficients can vary from the simplest ones like Haar, over Daubechies up to those like Quadratic Spline, having different vector lengths and, usually, floating point interpretation.

The Haar wavelet is considered to be the simplest one with two coefficients per filter:

$$L_d=[1/\sqrt{2},\ 1/\sqrt{2}],\ H_d=[1/\sqrt{2},-1/\sqrt{2}]\,, \tag{4}$$

$$L_r=[\sqrt{2}/2,\sqrt{2}/2]\,,\ H_r=[-\sqrt{2}/2,\sqrt{2}/2]\,. \tag{5}$$

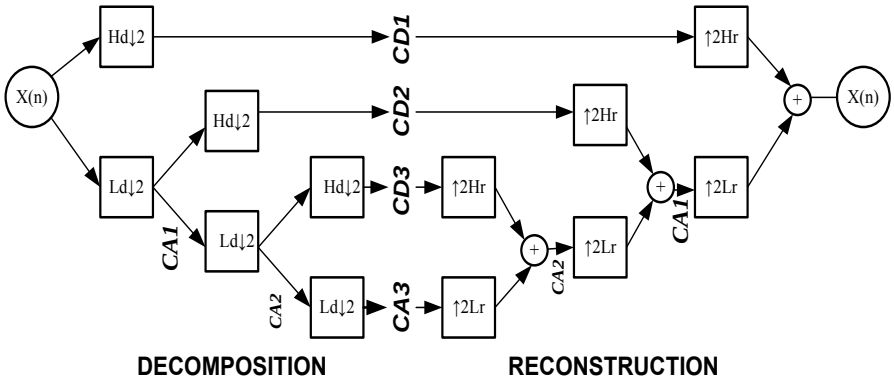


Fig. 1. Wavelet decomposition and reconstruction scheme

Haar transform (HT) has a number of advantages; it is (i) conceptually simple, (ii) fast, (iii) memory efficient, since it can be calculated in a place without a temporary array. Also, it is reversible without the edge effect that can be of a problem with some other WTs. But, this transform has several limitations, which can be of a problem in some applications, mainly in signal compression and noise removal from relatively high speed signals like audio or video. But, in the case of biomedical signals this is not an issue.

3.2. Integer-Based Optimization

Although very simple in its nature, HT is still complicated for implementation on low-complexity calculation devices like MCs. However, it can be generalized to an integer version. A technique proposed in [18] is in the form of S Transform (ST), whose Forward (FST) and Reverse (RST) versions are defined as:

$$CA_1[n] = \left\lfloor \frac{1}{2}X[2n] + \frac{1}{2}X[2n+1] \right\rfloor , \quad (6)$$

$$CD_1[n] = X[2n] - X[2n+1] , \quad (7)$$

$$X[2n] = CA_1[n] + \left\lfloor \frac{CD_1[n]+1}{2} \right\rfloor , \quad (8)$$

$$X[2n+1] = CA_1[n] - \left\lfloor \frac{CD_1[n]}{2} \right\rfloor , \quad (9)$$

where $\lfloor \ \rfloor$ denotes rounding operator. Because $\left\lfloor \frac{x}{2} \right\rfloor = x \gg 1$, FST and RST can be computed by mere adder-subtractor and shifter, what is, in practice, a key advantage.

3.3. WT-Based QRS Detection

WT is capable to distinguish the QRS-complexes within the ECG signal by implementing Mallat's decomposition scheme. $CD_i(n)$ coefficients across the scales show that the peak of the QRS complexes corresponds to the zero crossing (ZC) between two modulus maxima within the coefficients $CD_i(n)$ [19]. Fig. 2 illustrates the decomposition of discrete ECG signal $X(n)$ up to the 4th level, $CD_1(n)$, $CD_2(n)$, $CD_3(n)$ and $CD_4(n)$, by using above defined FST. For each decomposition level, the QRS complex produces two modulus maxima (*min* and *max*) with opposite signs and ZC between, see diagram CD_4.

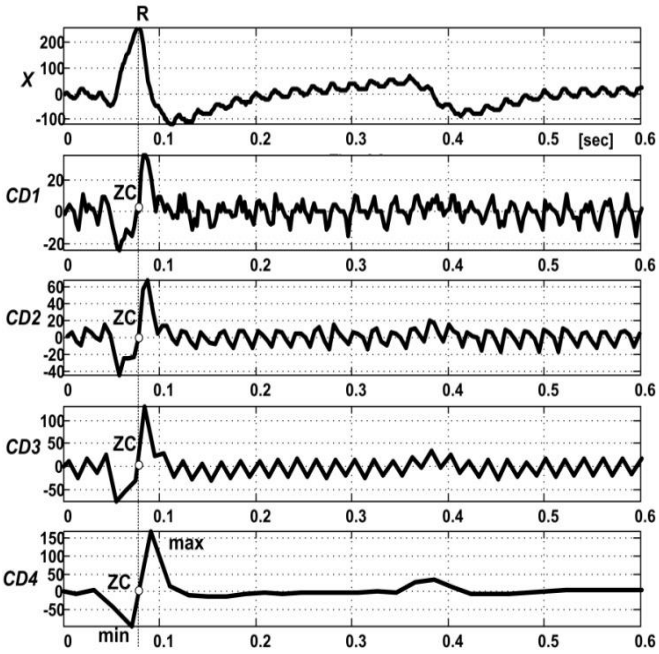


Fig. 2. QRS detection using wavelet decomposition based on FST. Signal $X[n]$ is sampled by 800 Hz. $CD_i(n)$ are the details after i[th] decomposition level

The method is very robust and allows direct application over raw ECG data. The frequency domain filtering is performed implicitly by computing the coefficients which is an additional positive feature, very useful in QRS detection. As can be observed, Fig. 2, the original signal becomes practically clear from 4th decomposition level.

Often, the modulus maxima (*min* and *max*) are found by thresholding techniques where the threshold *Tr* varies from one scale to another. For example, the thresholds can be calculated by Root Mean Square (RMS) function, as Tr=RMS($CD_i(n)$) for i=1,2 and 3 and Tr =0.5RMS($CD_i(n)$) for i=4, or by Maximum or Mean functions, Tr =MAX($CD_i(n)$) or Tr =MEAN($CD_i(n)$ [19].

In practice, the selection of the most suitable decomposition level/levels is of a challenge. The most of the energy of QRS complex lies between 3 Hz and 40 Hz. Translated to WT, it means somewhere between scales 2^3 and 2^4, with the largest at 2^4. The energy of motion artifacts and baseline wander (i.e. noise) increases for the scales greater than 2^5. Article [20] states that most energies of a typical QRS-complex are at scales 2^3 and 2^4, and the energy at scale 2^3 is the largest. According to [21], for QRS-complex with high frequency components, the energy at scale 2^2 is larger than that at scale 2^3 and authors recommend mainly the scales 2^3 to 2^4 for satisfactory detection.

Another complication is the acquisition of certain thresholds for finding the modulus maxima, because the values of thresholds differ, usually, from one level to another. The mentioned restrictions and complications confine the method to off-line use and put heavy demand on the computing resources.

3.4. Wavelet-Based Denoising

WT should be effectively used in signal filtering, here known as “de-noising”, especially in the elimination of high frequency and white noise [22]. “De-noising” consists of three successive procedures: decomposition, thresholding and signal reconstruction, Fig. 3a. Firstly, the wavelet transform is derived to a chosen level *N*. Secondly, the detail coefficients from level 1 to *N* are thresholded. Lastly, the original signal is synthesized using the altered detail coefficients from level 1 to *N* and approximation coefficients of level *N*.

There are several methods to define a threshold for the purpose of “de-noising”: global thresholding, where one threshold T_{hr} exists for all samples under consideration and level-based thresholding, where the vector of 2^N length, $T_{hr}(1..2^N)$, is used as a threshold [3]. Fig 3b. shows the case of global thresholding applied to the approximation coefficients of 4^{th} level and detailed coefficients of 1^{st} 2^{nd}, 3^{rd} and 4^{th} levels.

From another point of view, thresholding can be either soft or hard [3]. Hard thresholding zeroes out all the values smaller than T_{hr}. Soft thresholding does the same thing, and apart from that, subtracts T_{hr} from the values larger than T_{hr}. In the contrast to hard thresholding, soft thresholding causes no discontinuities in the resulting signal. Fig. 3b shows the effect of the wavelet-based filtering for ECG signal. The signal $X(n)$ is decomposed by FST till 4^{th} level, then thresholded by hard threshold $T_{hr}=0.23$ *V* and lastly reconstructed by RST. As can be seen, the reconstructed, filtered, signal $X'(n)$ is obtained from only 2.5% of nonzero coefficients.

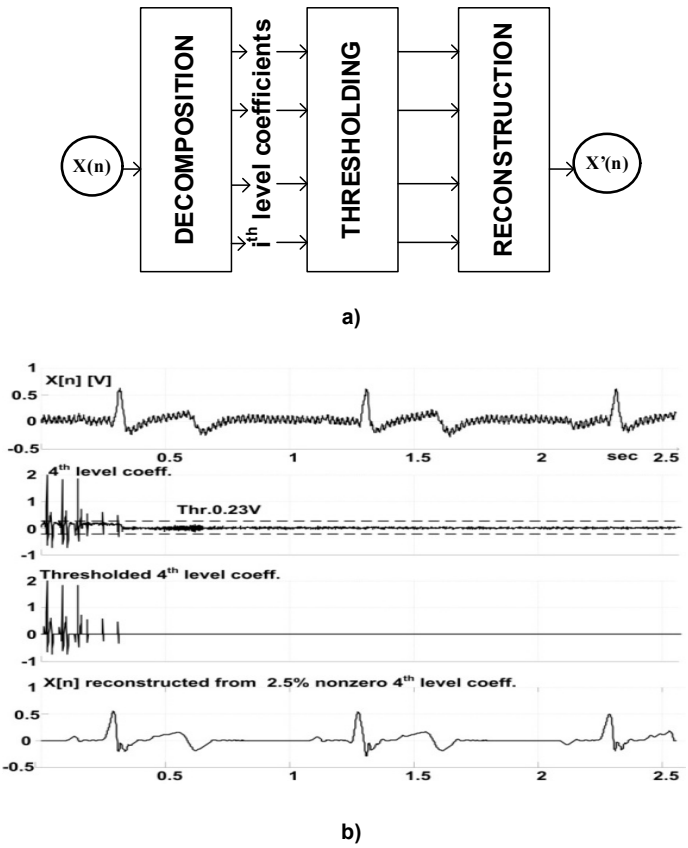


Fig. 3. a) Denoising steps, b) Effect on real ECG signal

4. Embedded Implementation

For the purpose of biomedical processing, the optimized QRS detection and "de-noising" algorithms are implemented in MSP430F169 microcontroller from MSP430 family, Texas Instruments TI [6]. It is a family of ultralow power microcontrollers optimized for using in portable battery powered devices like medical ones. The MSP430F169 has 16-bit RISC CPU, 16-bit registers, two 16-bit timers, fast 12-bit A/D converter with 8 external input channels, dual 12-bit D/A converter, USART, I2C, DMA, and 48 I/O pins, etc.

On-chip architecture for QRS detection is shown in Fig. 4. The analog ECG signal is fed to the channel A1 of internal ADC. After digitalization and processing in real-time, the output signals are generated in different forms: analog form of details $CD_N(n)$ and $CD_{N-1}(n)$ through the pins P6.6 and P6.7;

pulse form of RR intervals on P1.0 and string (ASCII) form of RR intervals through the USART's TX pin. The RR intervals are distances between QRS complexes, given in ms.

As it is mentioned in Section 2.3, the wavelet decomposition by itself presents a good noise filter used in QRS detection. "De-noising" technique, whose algorithmic steps are elaborated in Section 2, is an additional way to use wavelets as a filter. It is proved, in practice, as very effective tool for signal filtering. Fig. 5 presents wavelet based architecture for "de-noising", implemented in a single MC. The input signal is fed to A1 input of ADC, digitalized, decomposed by FST, thresholded, and finally reconstructed by RST. After reconstruction it is returned to analog form by DAC, see Fig.5 pin P6.7. Overall filtering process is performed in real-time.

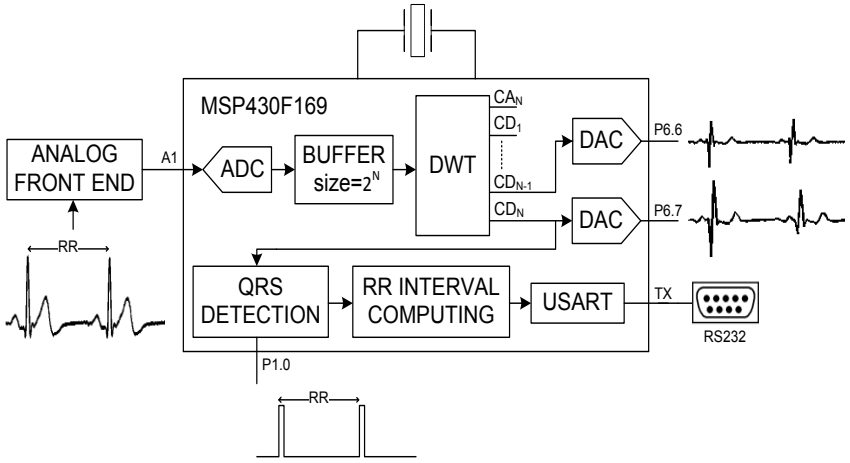


Fig. 4. MC architecture for QRS detection

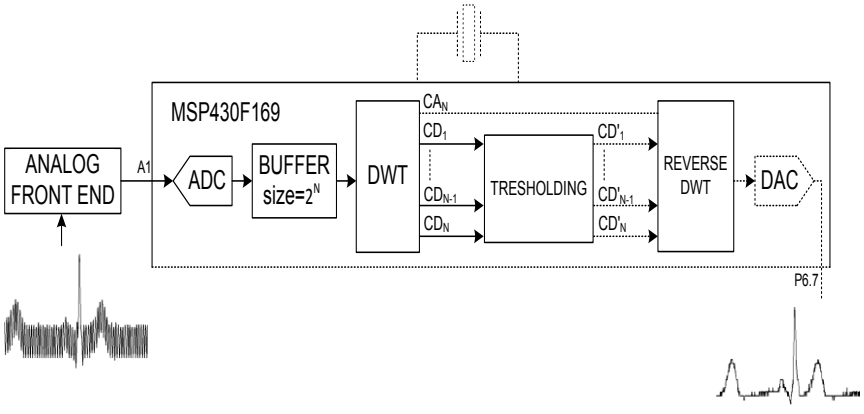


Fig. 5. MC architecture for denoising

The real-time implementation of forward and reverse wavelet transform is done through the FST and RST, because of their simplicity and fast calculation. Before processing, the signal is digitalized by 12-bit A/D converter. The sampling frequency is set at 800 Hz for QRS detection and at 762 Hz for denoising. The A/D conversion is performed in an interrupt routine. Between the interrupts, the MSP430 MC uses a low-power operating mode.

In the case of QRS detection, after A/D conversion, each sample is stored in a circular buffer of 2^N length, where N represents the number of decomposition levels. When the buffer is filled, the FST is calculated, while the buffer continues to accept new samples. In this research, the decomposition is done till $CD_4(n)$. Then, the $CD_4(n)$ are examined on ZC using negative and positive modulus maxima which are isolated by adaptive thresholding technique. Namely, five successive vectors of 50 CD_4 coefficients are examined. For each of them the maximum $M_jmax=max(CD_4(1..50))$ and minimum $M_jmin=min(CD_4(1..50))$ are determined, M_jmax, and M_jmin, $j=1..5$. Then the negative (T_1) and positive (T_2) thresholds are defined as:

$$T_1 = \frac{1}{4}\left(\frac{1}{5}\sum_{j=1}^{5} M_j\, min\right), \quad (10)$$

$$T_2 = \frac{1}{4}\left(\frac{1}{5}\sum_{j=1}^{5} M_j\, max\right). \quad (11)$$

Further, the process repeats with values from four old vectors and one new vector. ZC is detected by finding the coefficients associated to the condition $CD_4(n-1)<0$ and $CD_4(n)> 0$.

Detailed algorithm is given in Fig. 6. After computing a new CD_4 coefficient, check is performed to see whether that coefficient presents 50^{th} or not? If yes, the T_1 and T_2 thresholds are set. Then, searching for the negative modulus begins and in case of finding it search for ZC begins. After finding negative modulus and ZC, the algorithm is continuing to search for the positive modulus. If the negative modulus, ZC and the positive modulus are detected successively, then the QRS complex is detected and the algorithm starts to search for a new QRS complex.

In the case of "de-noising", the thresholding is implemented to each decomposition level. The detailed coefficients, whose absolute values are not greater than the threshold, are set to zero. For every decomposition level there is a separate adaptive threshold. For i^{th} (i=1..4) level, ten successive vectors v of W_i (i=1..4) coefficients, $v_{i,j}[1..W_i]$ (i=1..4, j=1..10) are taken in consideration. For each of them, the maximal value $A_{i,j}max=max(v_{i,j}[1..W_i])$ is found and stored in memory. Then, the adaptive threshold for i^{th} level, T_i, is calculated as average of the ten maximal values from that level, which is defined as:

$$T_i = \frac{1}{10}\left(\sum_{j=1}^{10} A_{i,j}\, max\right) \quad (12)$$

In order to maintain adaptability of the system for “de-noising”, calculation of the threshold continues with nine old maximal values and one new, which is found within a new vector of CD_i coefficients.

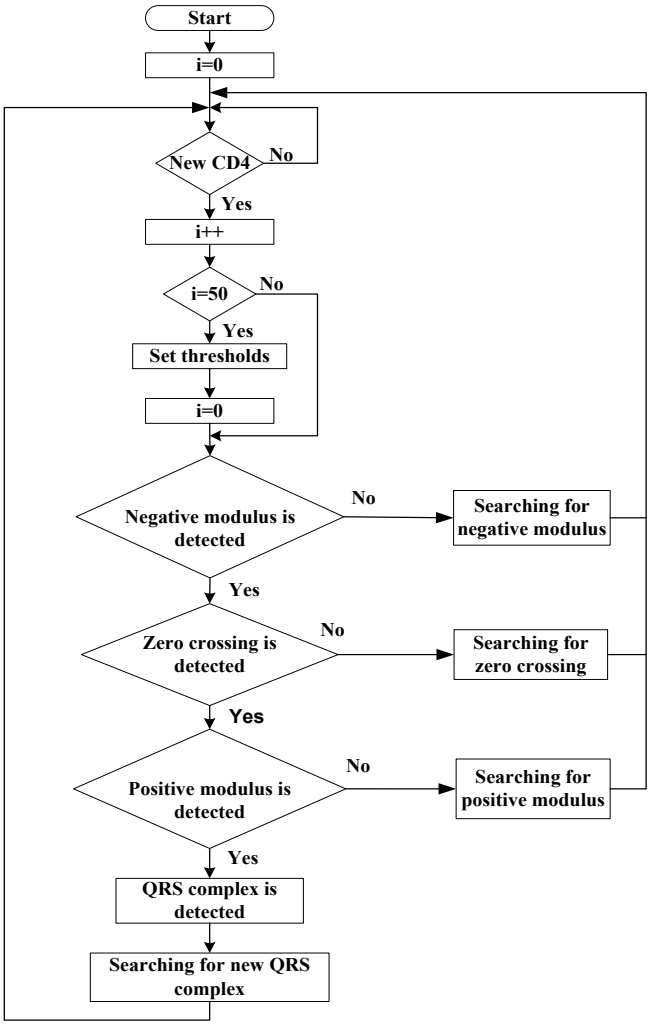


Fig. 6. The Algorithm for QRS detection which is implemented on MSP430 MC.

5. Results

For purpose of MC implementation and testing, the above presented algorithms for QRS detection and "de-noising" are developed in C code using IAR Embedded Workbench Compiler and then uploaded to MSP430F169 chip, through the Olimex MSP430-P169 development board. The verification of operation and necessary measurements are performed by tool-set consisting of PC, ELVIS II$^+$ NI Platform [23] and digital oscilloscope AGILENT DSO3120A. Designed, LabView Virtual Instrument (VI) read ECG signals from corresponding MIT-BIH files or PPG signals from laboratory files and convert them into analog form via ELVIS II$^+$ platform.

MSP430 chip accepts the emulated signals, performs FST and RST, QRS detection or "de-noising" in real-time. It returns the different analog or digital signals on output pins depending on the running program; $CD_4(n)$ and $CD_3(n)$ in the analog form; RR intervals in pulse (digital) form and RS232 RR intervals in ASCII string form. These signals are observed by oscilloscope or by terminal emulator in case of serial RS232 transmission. Further, the qualitative and quantitative analyses are performed.

5.1. Qualitative Analysis

This analysis is mainly performed by on-chip measurements. MC is configured to work in three modes, wavelet decomposition, QRS detector with digital outputs and "de-noising".

In the first mode, the emulated ECG signals are fed to the A/D input A1, digitalized and processed generating analog signals, $CD_3(n)$ *and* $CD_4(n)$ equivalents, on *D/A pins* P6.6 and P6.7, see Fig. 4. Simultaneously, the input and output waveforms are traced by digital oscilloscope. Then, the same ECG signals are processed by MATLAB, off-line, and results are plotted. For illustration, Fig. 7 shows the oscillographs and MATLAB plots of the input ECG signal and corresponding $CD_4(n)$ coefficients. As seen, the waveforms in Fig. 7 b) and Fig. 7 c) match very well. Note that the oscillograph amplitude and time division are printed in legend, bellow waveforms, as example, CH1 200 mV/div, 200.0 ms/div.

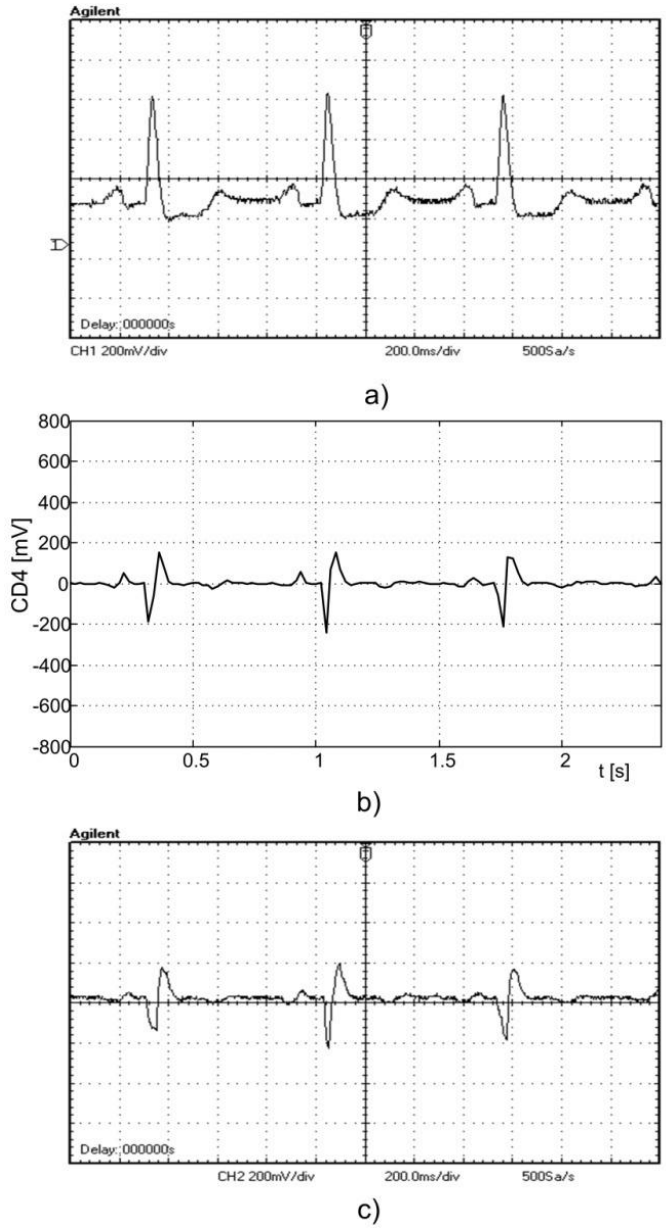


Fig. 7. FST calculated by MATLAB, off-line, and by MSP430F169, on-line. a) the oscillograph of the original ECG signal, b) $CD_4(n)$ coefficients plotted by MATLAB, c) oscillograph of $CD_4(n)$ coefficients, recorded on P6.7 pin. The sampling frequency was 800 Hz

In the second mode, the ECG signal is fed to the A/D pin A1, see Fig. 4. The MC performs QRS detection in real time and generates the RR impulses

(pin P1.0), whose positions correspond to the QRS complexes. The time distance between two successive impulses gives a RR interval in ms. Fig. 8 shows the oscillographs of original signal (up) and RR intervals (down). For example, the distance between 1st and 2nd impulse is 580 ms and between 2nd and 3rd is 560 ms that corresponds to the heart rates of 60*1/0.58=103 and 60*1/0.56=107 beats/pm, pm=per minute, indicating an effect of heart rate variability. As can be seen, the generated RR impulses are delayed, shifted, in relation to input signal, for about 50 ms.

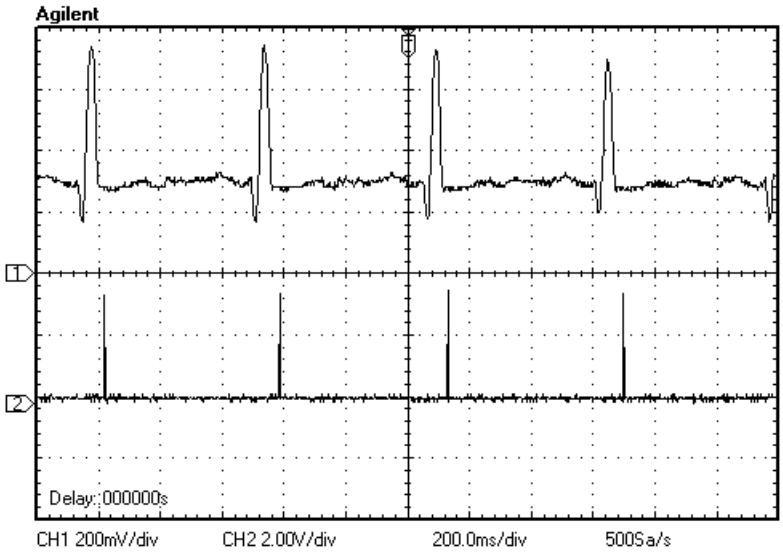


Fig. 8. ECG signal with QRS complexes (up) and RR impulses (down) obtained as a result of QRS detection. The sampling frequency was 800 Hz

Third mode is related to real-time “de-noising”, see Fig. 5. Analog forms of ECG and PPG signals, corrupted by 50 Hz or white noise, are fed to the A/D pin A1. The MC digitalize signal, runs “de-noising” code and, in real time, generates the filtered analog signals, D/A pin P6.7. Fig. 9 illustrates the situation with ECG signal corrupted by 50 Hz noise, while Fig. 10 shows filtering results against white noise. Fig. 11 illustrates the case of PPG signal corrupted by 50 Hz noise. The sampling frequency is 762 Hz and filtered signal is delayed for 40 ms. As can be seen, in all cases, the input signals are well filtered after passing “de-noising” code.

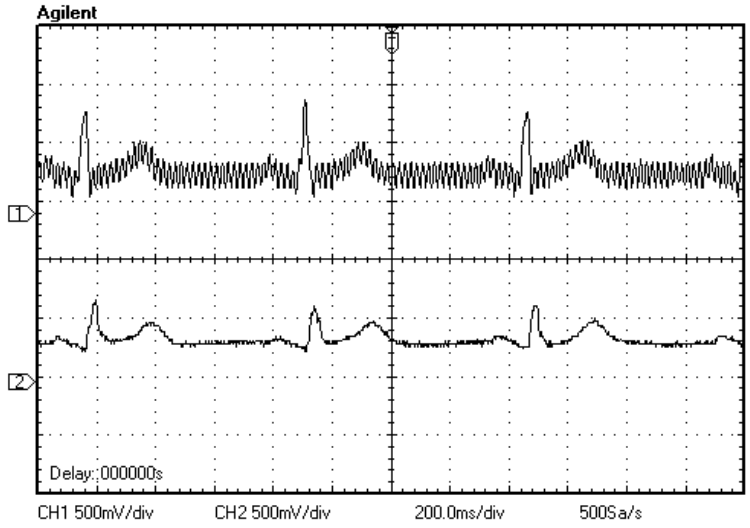


Fig. 9. ECG signal corrupted with 50 Hz noise (up) and filtering output (down)

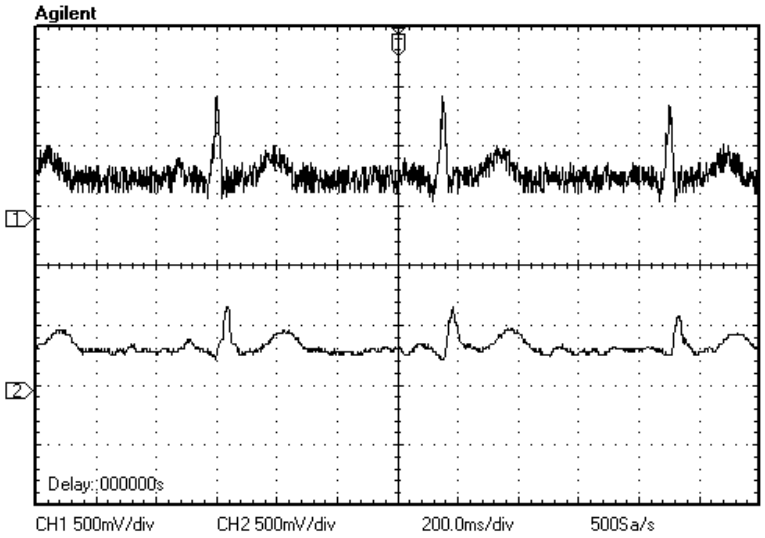


Fig. 10. ECG signal corrupted with white noise (up) and filtering output (down)

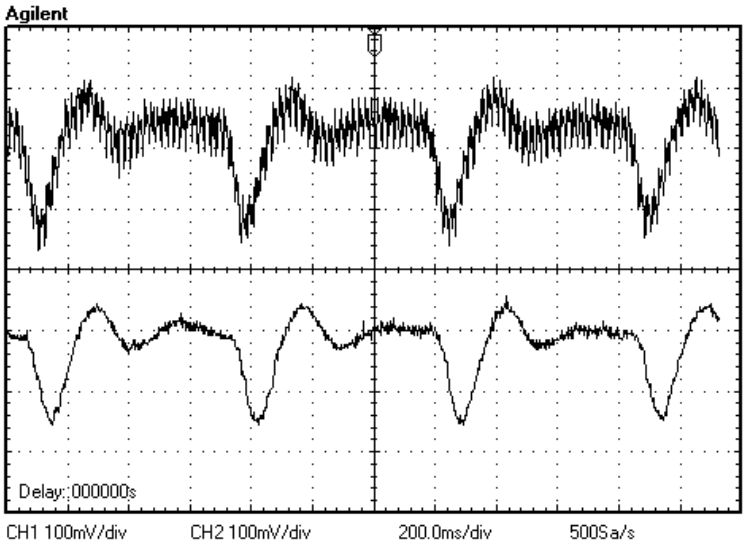


Fig. 11. PPG signal corrupted with 50 Hz noise (up) and filtering output (down)

5.2. Quantitative Analysis

In addition to the qualitative analysis, the proposed algorithms are evaluated against five (5) quantitative criteria: calculation time, data memory occupation, power consumption, detection accuracy and SNR. In all cases the MC is clocked by 0.75 MHz and powered by 3.3 V.

The calculation time is considered for floating point forward and inverse Haar Transformations (HTs) and proposed fixed point FST and RST. Table 1 gives the results. It is evident that fixed point implementation is more than two times faster for case of forward transform and more than three times faster for case of inverse transform. This fact allows MC to perform real time sampling and processing till 800 Hz, up to 4 levels, what significantly improves the quality of acquisition as well as detection accuracy.

Table 2 gives the memory occupation for floating point and fixed point implementations. And here, the difference is about two times in favor of fixed point. It should be noted that QRS detector implemented by FST occupies in total 224 bytes of DATA memory (+ 44 absolute), 39 bytes of CONST memory and 2 022 bytes of CODE memory. For the case of “de-noising” it is 302 bytes of DATA memory (+ 33 absolute) and 1902 bytes of CODE memory.

Table 1. Calculation times for floating and fixed point transforms

# of decomposition levels	FST [ms]	RST [ms]	Forward HT [ms]	Inverse HT [ms]
4	2,35	2,23	5,82	6,90
5	3,96	4,37	11,86	13,99
6	6,93	8,63	23,91	28,14
7	12,64	17,10	47,99	56,41

Table 2. Memory occupation, DATA MEMORY, RAM, for floating and fixed point transforms

# of decomposition levels	FST [bytes]	RST [bytes]	Forward HT [bytes]	Inverse HT [bytes]
4	74	74	138	138
5	138	138	266	266
6	266	266	522	522
7	522	522	1034	1034

By its nature MSP430x is an ultra low power controller. Additionally, the integer point optimization slightly decreases consumption. QRS detector and filter, implemented in this arithmetic, consumed 319 μA and 315 μA that is about 5% less than in case of floating point calculations, 336 μA, 332 μA.

In order to verify the QRS detection accuracy, the 11.094 heart beats within five characteristic files are observed (MIT-BIH Records 101, 103, 202, 230, 234). The particular detection error rate for each record, DER_i, is defined as:

$$DER_i[\%] = 100\left(1 - \frac{NFP + NFN}{TN}\right) \quad (\mathbf{13})$$

where are: *NFP* - number of false positives in $X_i[n]$, *NFN* - number of false negatives in $X_i[n]$ and *TN* - total number of QRS complexes in $X_i[n]$. The averaged accuracy is defined as:

$$ADER[\%] = \frac{1}{5}\sum_{i=1}^{i=5} DER_i\,[\%]\ . \quad (\mathbf{14})$$

First, the files are passed through the wavelet based QRS detector realized in MATLAB by algorithm structure and method of modulus maxima given in [24] with distinction that Mexican hat wavelet is replaced with Haar. Then, the analog ECG signals are feed to the proposed MC's QRS detector. ASCII forms of RR intervals are collected by terminal emulator and then statistically analyzed by MATLAB. The averaged accuracies were 99.47% and 99.06%, respectively. Obviously, the proposed MC detector decreases accuracy for -0.41% what can be considered as negligible.

In order to quantitative estimate "de-noising" technique, the output *SNR*, SNR_o, is considered for initial value of *SNR*, SNR_i :

$$SNR_o = 10log\frac{\sum_{i=1}^{N} X(i)^2}{\sum_{i=1}^{N}(X(i) - X_r(i))^2}, \quad (\mathbf{15})$$

$$SNR_i = 10log\frac{\sum_{i=1}^{N} X(i)^2}{\sum_{i=1}^{N} n(i)^2}, \quad (\mathbf{16})$$

where, *X(i)* is the original signal, $X_r(i)$ is "de-noised" signal, *n(i)* noise signal and *N* is the length of the signals.

The ECG signals from above MIT-BIH records are corrupted by 50 Hz noise of different amplitudes and passed through the MATLAB codes of proposed MC's "de-noising" algorithm and algorithm based on HT with hard thresholding from [22]. The results are shown in Table 3.

Table 3. SNR_o values for "de-noising" algorithms

SNR_i	30dB	20dB	10dB	5dB
SNR_o – HT	32.5601	23.2341	13.5353	8.6026
SNRo – Proposed alg.	31.9473	22.7246	13.2348	8.3626
Improvement - Degradation [%]	-1.8821	-2.1929	-2.2201	-2.7899

As can be noted, the classical HT with hard thresholding has better *SNRo*. However, the degradation for proposed algorithm, even in the worst case, is negligible, less than 2.8%.

6. Conclusion

Wavelet transforms can be successfully used to solve many tasks in biomedical signal processing. After certain optimizations in the terms of fixed point arithmetic, they can be implemented in low-cost general purpose microcontrollers. Case studies for real-time QRS detection and ECG and PPG "de-noising", implemented in MSP430F169, are presented. The benefits are obvious, 800 Hz sampling rate, 2-4 times faster calculation, less than 500 bytes of data memory occupation, 1 mW power consumption, 99.06% detection accuracy, 5% decreased power consumption and satisfied SNR. The degradations are negligible about -0.41% in accuracy and -2.8%, in SNR. The same approach can be applied with other signals where the embedded implementation of wavelets can be beneficial.

Acknowledgment. This paper presents a part of the research performed in the projects: "Development and implementation of embedded systems for medical applications", MESI, supported by Ministry of Science of Montenegro, "Application of Biomedical Engineering in Preclinical and Clinical Practice", III-41007, supported by

the Serbian Ministry of Science and Technology and TEMPUS, 530417-TEMPUS-1-2012-1-UK-TEMPUS-JPCR, "Studies in Bioengineering and Medical Informatics", supported by EU Commission. The authors are grateful for their support.

References

1. Graps, A.: An Introduction to Wavelets. Computing in Science and Engineering, Vol. 2, No. 2, IEEE press, 50-61. (1995)
2. Makris, C.: Wavelet trees: a survey. Computer Science and Information Systems, Vol. 9, No. 2, 585-625. (2012)
3. Merry, R.J.E.: Wavelet theory and applications: a literature study. Technische Universiteit Eindhoven (Eindhoven), DCT 2005.053. (2005) [Online]. Available: http://alexandria.tue.nl/repository/books/612762.pdf
4. Zhao, J., Zhang, Z., Han, S., Qu, C., Yuan, Z., Zhang, D.: SVM Based Forest Fire Detection Using Static and Dynamic Features. Computer Science and Information Systems, Vol. 8, No. 3, 821-841. (2011)
5. Addison, P.S.: Wavelet transforms and the ECG: a review. Physiological Measurements, Vol. 26, 155–199. (2005)
6. Texas Instruments Home Page, [Online]. Available: http://www.ti.com/lsds/ti/microcontroller/16-bit_msp430/overview.page?DCMP=MCU_other&HQS=msp430, (January 2012)
7. Vogel, S., Hulsbusch, M., Hennig, T., Blazek, V., Leonhardt, S.: In-Ear Vital Signs Monitoring Using a Novel Microoptic Reflective Sensor. IEEE Trans Inf Technol Biomed, Vol. 13, No. 6, 882-889. (2009)
8. Pantelopoulos, A., Bourbakis, N.G.: A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 40, Issue 1, 1-22. (2010)
9. Stojanovic, R., Karadaglic, D.: A LED–LED-based photoplethysmography sensor. Physiol. Meas. Vol. 28, 19-27. (2007)
10. Phyu, M. W., Zheng, Y., Zhao, B., Liu, X., Wang, Y. S.: A Real-Time ECG QRS Detection ASIC Based on Wavelet Multiscale Analysis. In Proceedings of the Solid-State Circuits Conference, A-SSCC 2009*.,* 293-296. (2009)
11. Hoang, T. T., Son, J. P., Kang, Y. R., Kim, C. R., Chung, H. Y., Kim, S. W.: A Low Complexity, Low Power, Programmable QRS Detector Based on Wavelet Transform for Implantable Pacemaker. In Proceedings of the 19th IEEE System on Chip Conference (SOCC), Texas, USA, 160-163. (2006)
12. Chio, I. I., Mang, I. V., Peng, U. M.: ECG QRS Complex Detection with Programmable Hardware. In Proceedings *of* the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*,* Vancouver, British Columbia, Canada, 2920-2923. (2008)
13. Stojanović, R., Karadaglić, D., Mirković, M., Milošević, D.: A FPGA system for QRS complex detection based on Integer Wavelet Transform. Measurement Science Review, Vol. 11, Issue 4, 131-138. (June 2011)
14. Bahoura, M., Hassani, M., Hubin, M.: DSP Implementation of Wavelet Transform for Real Time ECG Wave Forms Detection And Heart Rate Analysis. Computer Methods and Programs in Biomedicine, Vol. 52, 35-44. (1997)
15. Rudnicki, M., Strumillo, P.: A Real-Time Adaptive Wavelet Transform-Based QRS Complex Detector. In proceeding of 8th International Conference on

Adaptive and Natural Computing Algorithms, ICANNGA 2007, Proceedings, Part II, Warsaw, Poland, 281-289. (2007)
16. Kumar, P., Jain, M., Chandra, S.: Low Cost, Low Power QRS Detection Module Using PIC. In Proceedings of the 2011 International Conference on Communication Systems and Network Technologies, Katra, Jammu, India, 414-418. (2011)
17. Mallat, S.G.: A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 11, No. 7, 674-693. (1989)
18. Calderbank, A.R., Daubechies, I., Sweldens, W., Yeod, B.L.: Wavelet Transforms That Map Integers to Integers. Applied and Computational Harmonic Analysis, Vol. 5, Issue 3, 332-369. (1998)
19. Li, C., Zheng, C., Tai, C.: Detection of ECG characteristic points using wavelet transform. IEEE Transactions on Biomedical Engineering, Vol. 42, Issue 1, 21-28. (1995)
20. Almeida, R., Martinez, J.P., Olmos, S., Rocha, A.P., Laguna, P.: A wavelet-based ECG delineator: Evaluation on standard databases. IEEE Transactions on Biomedical Engineering, Vol. 51, No. 4, 570-581. (2004)
21. Thakor, N.V., Webster, J.G., Tompkins, W.J.: Estimation of QRS complex power spectra for design of a QRS filter. IEEE Trans Biomed Eng., Vol. 31, Issue 11, 702-706. (1984)
22. Šindelářová, I., Ptáček, J., Procházka, A.: Wavelet Transform in Signal De-Noising. Proc. of the 5th Int. Conf. Process Control 2002, R190/1-7. (2002)
23. National Instruments, NI ELVIS II Series Specifications, [Online]. Available: http://www.ni.com/pdf/manuals/372590b.pdf, (January 2012)
24. Romero Legarreta, I., Addison, P.S., Grubb, N.: R-wave Detection Using Continuous Wavelet Modulus Maxima. Computers in Cardiology, Vol. 30, 565−568. (2003)

Radovan Stojanović received his M.Sc. from University of Montenegro, in 1990, and Ph.D. from University of Patras, Greece, in 2001. He is currently associate professor at the University of Montenegro. His research interests include embedded systems, applied image and signal processing, instrumentation and measurements, industrial electronics, biomedical engineering. He is an author or cooauthor of more than 150 research papers as well as a coorinator of numerous international, bilateral and national projects. He is a member of the IEEE, associate fellow of IIAS, visiting researcher and lecturer at several EU universities and institutes end founder of Mediteranian Embedded Comupting (MECO) events.

Saša Knežević is currently M.Sc. student at Faculty of Electrical Engineering of University of Montenegro. He graduated in 2011 at the Department for Electronics at Faculty of Electrical Engineering of University of Montenegro. His research interests are embedded systems, CAD and software engineering.

Dejan Karadaglić, DPhil (Oxon), CEng, MIET, CPhys, MInstP, is a Lecturer at the School of Engineering and Built Environment, Glasgow Caledonian University. His research interests are focused at sensors and imaging area with broad range application, but primarily using optoelectronic techniques in biomedical engineering. He worked at the Universities of Oxford, St Andrews, Liverpool and Manchester in past, where participated in a number of leading-edge technology projects, and published a number of peer-reviewed publications.

Goran Devedžić is professor at Faculty of Engineering, University of Kragujevac, Serbia. His research interests focus on the advanced product and process development, industrial and medical application of soft computing techniques, and bioengineering. He has authored/co-authored more than 100 research papers, published in international and national journals or presented at international and national conferences, as well as three books on CAD/CAM technology and 3D product modeling.

Received: May 17, 2011; Accepted: November 23, 2012.

DOI:10.2298/CSIS120531014F

Biomechanical Modeling of Knee for Specific Patients with Chronic Anterior Cruciate Ligament Injury

Nenad Filipović[1,2], Velibor Isailovic[1,2], Dalibor Nikolić[1,2], Aleksandar Peulić[3], Nikola Mijailović[1,2], Suzana Petrović[1], Saša Cuković[1], Radun Vulović[1,2], Aleksandar Matić[3], Nebojša Zdravković[3], Goran Devedžić[1] and Branko Ristić[3]

[1] Faculty of Engineering, University of Kragujevac, Sestre Janjica 6,
34000 Kragujevac, Serbia
fica@kg.ac.rs
[2] Bioengineering Research and Development Center, Prvoslava Stojanovica 6,
34000 Kragujevac, Serbia
bioirc@kg.ac.rs
[3]Technical Faculty,
32000 Cacak, Serbia
[4] Medical Faculty, University of Kragujevac, Svetozara Markovica 69,
34000 Kragujevac, Serbia
branko.ristic@gmail.com

Abstract. In this study we modeled a patient specific 3D knee after anterior cruicate ligament (ACL) reconstruction. The purpose of the ACL reconstruction is to achieve stability in the entire range of motion of the knee and the establishment of the normal gait pattern. We present a new reconstruction technique that generates patient-specific 3D knee models from patient's magnetic resonant images (MRIs). The motion of the ACL reconstruction patients is measured by OptiTrack system with six infrared cameras. Finite element model of bones, cartilage and meniscus is used for determination stress and strain distribution at different body postures during gait analysis. It was observed that the maximum effective von Mises stress distribution up to 8 MPa occurred during 30% of the gait cycle on the meniscus. The biomechanical model of the knee joint during gait analysis can provide insight into the underlying mechanisms of knee function after ACL reconstruction.

Keywords: ACL reconstruction, knee motion, gait analysis, biomechanical finite element modeling.

1. Introduction

There is always a question what are dynamic loading conditions to which cartilage is exposed during daily activity. It is fundamental for diagnosing and treating joint disease, since dynamic loading affects the movement of tissue growth factors [1].

Interaction between several factors (anatomical, functional and biological) has influence on cartilage degeneration. Determining cartilage progression rate is based on defining abnormal loadings during gait cycle which contribute to cartilage wear.

Many researchers analyzed biomechanical models of the knee joint based on the finite element method. These models provide significant insight into the stress and strain distribution and contact kinematics at the knee joint [2], [3], [4], [5], [6] and have been used to investigate the effect of ligament injury [7], [8]. In these studies the knee joint was generally subjected to axial loads with the knee flexion angle fixed and subject-specific data were not used to define the joint geometry and loading conditions. To address these shortcomings, here, we propose a construction of subject-specific biomechanical model of the human knee joint by combining magnetic resonance imaging (MRI) of the knee joint, motion analysis measured with camera system and finite element analysis of subject-specific 3D knee models.

2. Related Work

Normal knee functions lie in complex relationship of the movement and stability. Knowing knee kinematics is of great importance for getting relevant knee functions information which can be used for improving treatment of the knee pathology.

Clinical and functional indicators of the surgery results of the anterior cruciate ligament show decrease of the tibial translation during gait activity in the postoperative period. Some studies show patients' ability to reduce tibial translation at the deficient knee although knee laxity is obvious. Reduction of the tibial translation is influenced by muscle activity. Primary task of the reconstruction surgery is to reduce translation of the tibia in the sagittal plane [9], [10], [11].

Tibial translation generally drives knee stability after anterior cruciate ligament reconstruction. The problem can still develop in spite of a decrease in excessive anteroposterior tibial translation after surgical procedure.

K. Manal et al. show that movement of the soft tissue of the lower limb could influence on the appearance of error during estimation of the tibial translation [12]. B. Gao et al. show that there exists significant change in the joint kinematics between deficient anterior cruciate ligament and healthy knee. After reconstructive surgery some differences corrected, but normal knee kinematics is not completely restored. In this study for measured

purposes we used method for gait analysis and optimization algorithm in order to reduce analysis errors caused by movement of the soft tissue [13].

According to the numerous studies during in vitro and in vivo experiments tibial translation along AP direction has been noticed in the case of the deficient anterior cruciate ligament knee, which confirms the findings in our study, shown at the Figure 3 [14], [15]. Maximal values of the tibial translation along ML, IS, and AP directions appear in the early stance phase. The ligament reconstruction surgery decreases tibial dislocation along all above mentioned directions.

The stability of the human knee joint is influenced by comprised elements such as ligaments, menisci and muscles. If deficient anterior cruciate ligament knee is not reconstructed, it indicates degenerative process on the cartilage [8], [9], [15].

In this paper we used a nonlinear porous finite element analysis for cartilage and meniscus and linear model for knee stability after anterior cruciate ligament reconstruction. It is very important to better understand cartilage and meniscus behavior to different loading condition. Many medical doctors found that the cartilage injury was most severe over the superficial zone of the posterior lateral tibia. It is impossible to measure injury in vivo patients even with today's state of the art for the image reconstruction methods. By comparing the computer simulation stress and strain cartilage and meniscus values we will be able to assess the severity of each patient's injury more accurately.

The paper is organized as following. We firstly present methods for experimental measurement, 3D image segmentation and reconstruction and finite element model of cartilage and meniscus. Then some results for coupled measurement and computational analysis are described. At the end some discussion and conclusion remarks are given.

3. Methods

3.1. Experimental measurements

Gait analysis was performed with nineteen adult men which are voluntarily participated in the experimental measurements. Subjects had a mean height of 183.33cm (S.D. 2.24), mean weight of 86kg (S.D. 3.48) and mean age of 29.89 years (S.D. 1.73). Subjects are recreational or professional sportsmen. Test analysis and surgery were performed at Clinical Centre Kragujevac, (Clinic for Orthopedics and Traumatology).

Kinematic data were collected with a three – dimensional (3D) motion analysis system (OptiTrack). This system consist of recording software ARENA and six infrared cameras (V100:R2) resolution 640x480 pixels with frame rate of 100 fps. Cameras were placed along a pathway. For defining

and processing kinematic data the global coordinate system was used because it is stationary, it does not depend on the subject and it is not influenced by marker's position. Global coordinate system was defined with z – axis coincidence with inferior - superior (IS) direction, x – axis coincidence with medial - lateral (ML) direction, and y – axis coincidence with anterior - posterior (AP) direction [8].

The study was performed in order to define kinematics data of the lower limb during performing gait activities in patients with deficient anterior cruciate ligament of the knee. Four passive reflective markers were placed at the anatomical landmarks of the lower extremity in order to minimize muscle activity. Landmarks were defined at the great trochanter region (GTR), at the femoral lateral epycondile (LEF), at the tuberosity of the tibia (TT), and in the region of the center of the ankle joint (CAJ) (Fig.1).

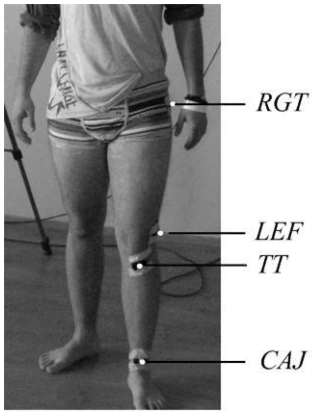


Fig. 1. The marker set used in the gait analysis experiment

Subjects performed normal walk at a self – selected speed along pathway about 5.00m. The day before surgery were recorded signals, first at the knee with deficient AC ligament, and then at the healthy knee. Every subject was asked to perform this task four times. Experiments are repeated after 15 days, and after 6 weeks. In this paper the results of the gait analysis after 6 weeks are shown.

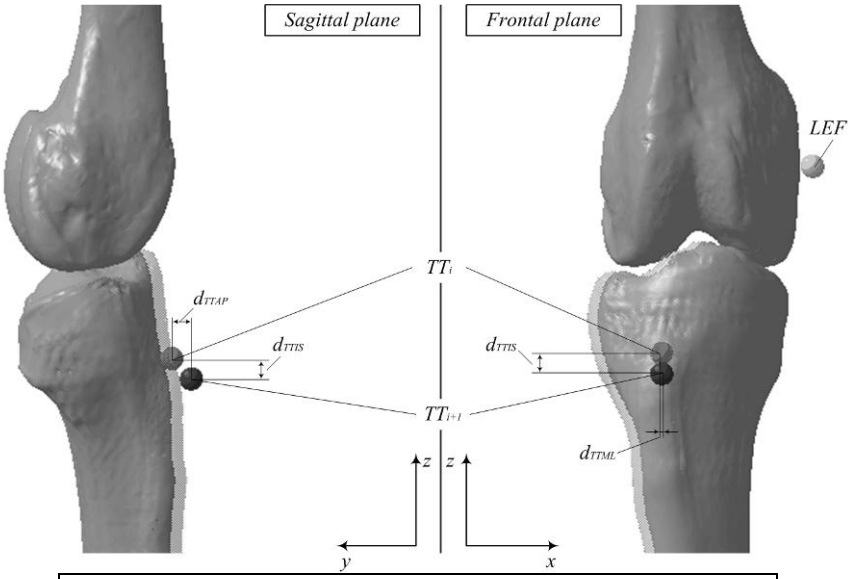
Since subjects had deficient anterior cruciate ligament of the knee, during walking (Fig.2), in one moment (point TTi) the knee is stable, but in the next moment there is a tibial shift (point TTi+1).

According to above mentioned, tibial dislocation was defined by successive calculating the affine coordinates along IS, ML, and AP directions [9] with equations (1)-(3):

$$d_{TTAP} = (TTAP)_{i+1} - (TTAP)_i , \quad (1)$$

$$d_{TTML} = (TTML)_{i+1} - (TTML)_i , \quad (2)$$

$$d_{TTIS} = (TTIS)_{i+1} - (TTIS)_i \,. \tag{3}$$



LEGEND:
LEF – lateral epycondil of the femur,
TTi – tuberosity of the tibia in i-th moment,
TTi+1 – tuberosity of the tibia in i+1-th moment,
dTTAP – tibial dislocation along AP direction,
dTTML – tibial dislocation along ML direction, and
dTTIS – tibial dislocation along IS direction.

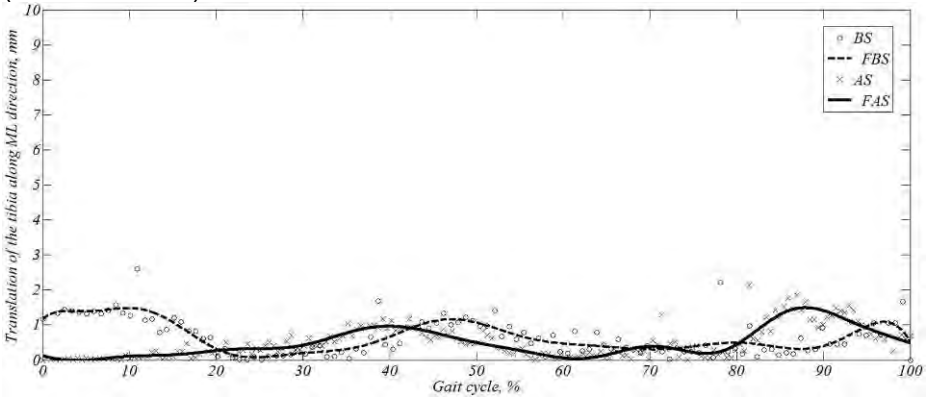
Fig. 2. Tibial dislocation along ML, IS, and AP directions

Using obtained data point we apply eight order Fourier series approximation to estimated the curves of tibia dislocation for a specific patient (Fig.3).

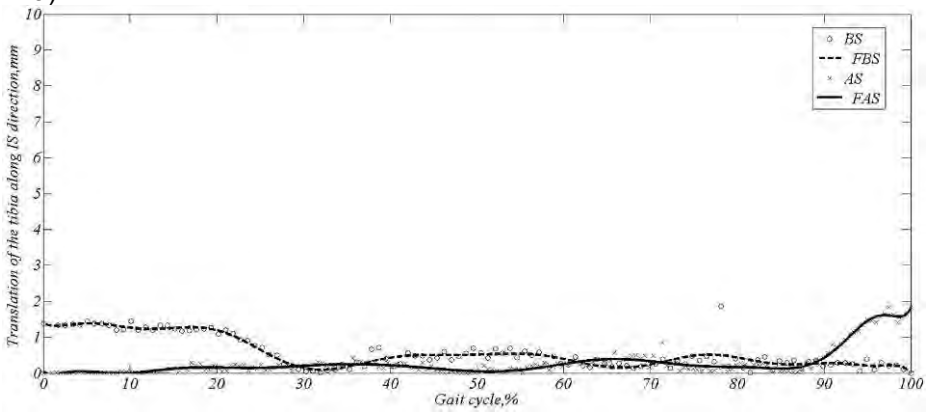
The diagrams in Figure 3 indicate that tibial dislocations along ML and IS directions before and after surgery are very small, and they do not have big influence on the knee stability [9], [10]. Mean value of the tibial translation before surgery along ML direction is:0.656 mm (S.D. 0.512 mm), and along IS direction is 0.553 mm (S.D. 0.445 mm). It can be seen that values of the tibial translation along ML and IS direction decreased after surgery. Mean values of these translations along ML direction is 0.387 mm (S.D. 0.324 mm), and along IS direction is 0.122 mm (S.D. 0.099 mm).

AP translation has big influence on the knee stability at knees with deficient anterior cruciate ligaments. The fitted curves on Figure 3 which describe AP translation have high amplitudes [9], [10]. In swing phase of the gait cycle which correspond to 40% of the horizontal axis can be seen sharp

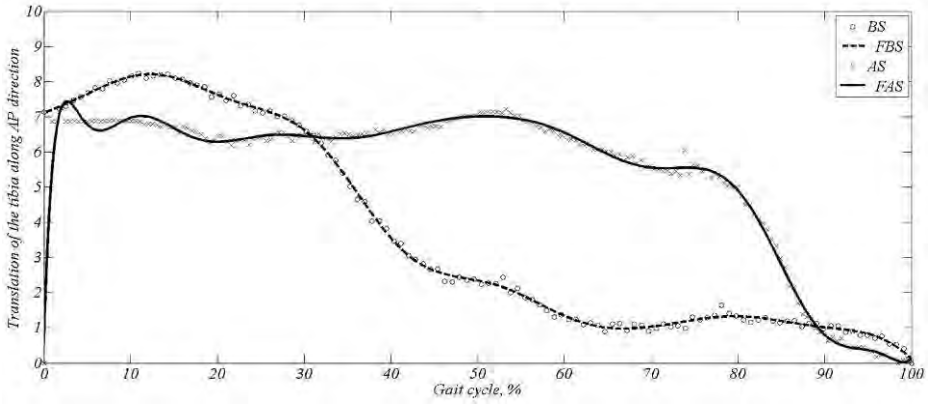
decline of the curve before surgery. Mean value of the AP translation before surgery is 4.543 mm (S.D. 3.658 mm). After ligament reconstruction, motion curve of the tibial translation has lower amplitudes and shows stability in swing phase. Mean value of the AP translation after surgery is 6.623 mm (S.D. 0.662 mm).



a)



b)



c)

LEGEND:
BS – Curve of the translation of the tibia before surgery, **FBS** – Fitted curve of the translation of the tibia before surgery, **AS** - Curve of the translation of the tibia after surgery, **FAS** – Fitted curve of the translation of the tibia after surgery.

Fig. 3. Translation of the tibia along: a) ML direction, b) IS direction, and c) AP direction

Student t – test was used for purpose of the statistical significance of the experimental results. It can be seen that there was significant difference in tibial translation along IS, ML, and AP directions in preoperational and post operational period for possible error $p< 0.01$ and for certainty of the $P>99\%$.

3.2. Computational method

We take geometry of the finite element model from MRI slices for a specific patient after surgery. Our in-house implementation includes an interface for users to adjust the position of the virtual cutting plane to better match with the MRI slices. A user can also make hand corrections on the knee contours after the automatic segmentation process finish. Four reflective markers at the anatomical landmarks of the lower extremity are detected on MRI 3D reconstruction object.

Figure 4 shows the interface of our knee segmentation system.

The algorithm for image segmentation and 3D object reconstruction from the MRI slices is following. Over all pixels a FE-mesh, initially uniform, is isotropically generated. We positioned the nodes of the FE-mesh at the centers of the existing voxels. This means that each FE overlaps 2x2 voxels in the 2D case or 2x2x2 voxels in the 3D case. The black circles represent nodes generated inside the object and the white circles denote nodes outside of the object. The nodes inside the object have a pixel or voxel value higher than the chosen threshold and the nodes outside the object have a value lower than the chosen threshold. Each FE node is assigned the grayscale value of the corresponding voxel. Note that the boundary between the black and white nodes is not smooth at this stage.

Because the FE-mesh is located on the surface boundary, some of its nodes (shown as the white circles) are on the outside of the object. Additionally, the grayscale pixel-values of those white nodes are lower than the chosen threshold value. By using a simple linear interpolation, we move these white nodes in the direction of the surface boundary toward the locations where the grayscale pixel-value would exactly match the threshold value. There are multiple methods available to move the nodes by linear interpolation. It is important to note that in some cases this linear interpolation might even move the node inside the object.

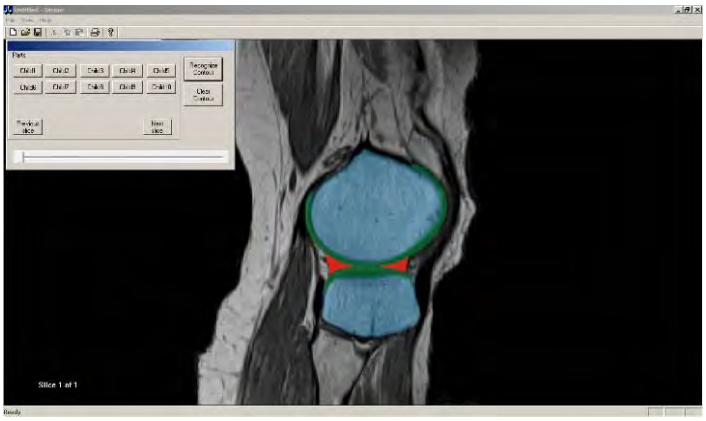


Fig. 4. Segmentation of the knee model from MRI slices. Sagittal view MRI of the left knee. The bone geometry, cartilages and meniscuses are digitized for 3D finite element model

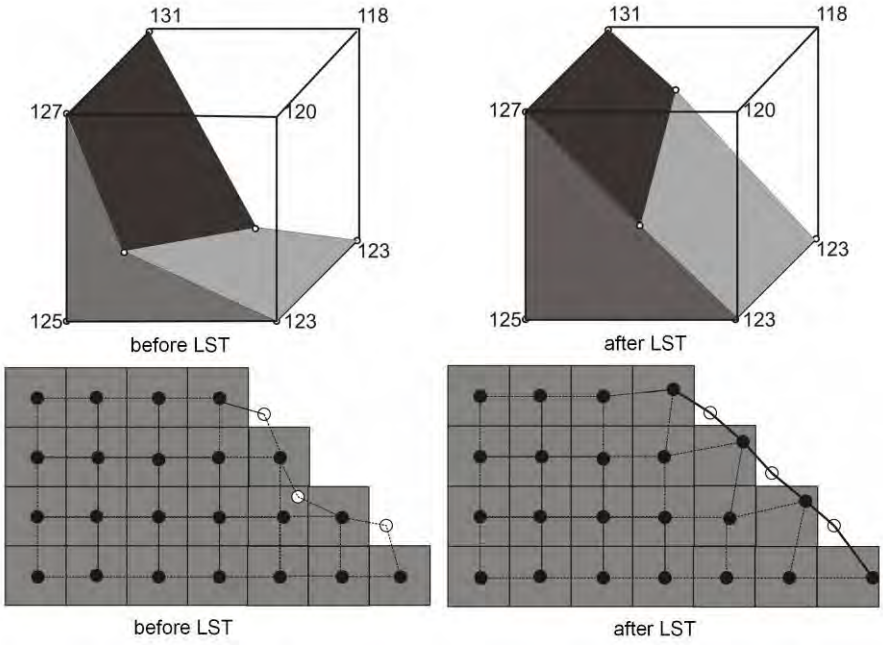


Fig. 5 Grid-Based Hexahedral Algorithm. 3D thresholded voxels (upper panel), and 2D representation of thresholded voxels (bottom panel). The grayscale values are kept in the map of voxel. Laplacian Smoothing Technique (LST). 3D and 2D before LFT (left panel), 3D and 2D after LFT (right panel)

The translation of the nodes may in some cases lead to a distorted (concave) FE-surface. The distortion of the FE-nodes can be evaluated with their Jacobian value. The Jacobian value is a matrix of the derivation of

global to local finite element interpolation function and the quality of any mesh can be directly evaluated by its Jacobian value. Distorted FEs, which are not suitable for subsequent numerical calculations, show a negative Jacobian. To optimize the Jacobian, we implemented the standard Laplacian Smoothing Technique (LST) [16]. The LST usually takes a few loops (repetitions of step ii) over all FEs to achieve positive Jacobian values for all FEs. The results of applying the LST for 3D and 2D cases are shown respectively in the right panels of Fig. 5 ("After LST") [17].

The finite element mesh is presented in Figure 6. Very fine mesh up to one million of finite elements is used.

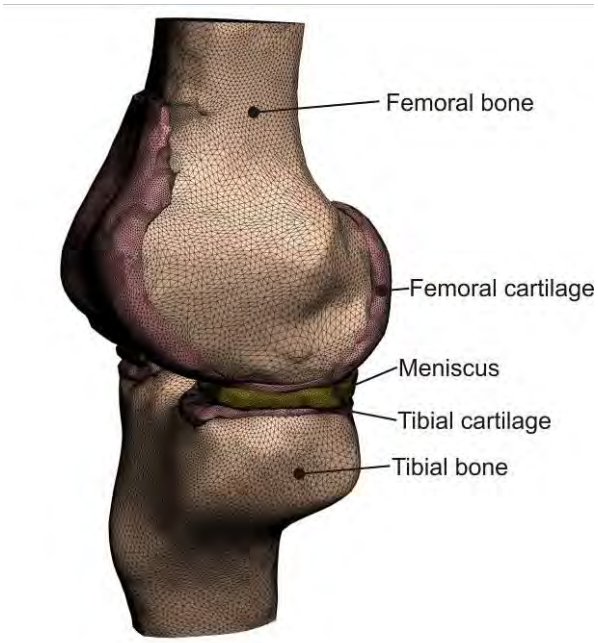


Fig 6. Finite element mesh for different knee segments

Some C pseudo code for contour recognition and 3d reconstruction code are given below. Detailed source is given in the Appendix.

```
void function mesh_generation
for (i = 1; i<=Num_voxel; i++) {
        set voxel=black;
if i is outside object
        set voxel=White;
}
void function surface_generation
for (i = 1; i<=Num_voxel; i++) {
        set Jacobian;
if Jacobian < 0
```

```
        use Lapacian_smoothing_algorithm;
}
void function Laplacian_ smoothing_algorithm
        set Node(x,y,z)
for (i = 1; i<=Num_node_el; i++) {
        X=Sum X(i);
        Y=Sum Y(i);
        Z=Sum Z(i);
        Node(x)=X/ Num_node_el;
        Node(y)=Y/ Num_node_el;
        Node(z)=Z/ Num_node_el;
}
```

For modeling of the cartilage and meniscus we implemented finite element formulation where the nodal variables are: displacements of solid, **U**; fluid pressure, **P**; Darcy's velocity, **Q;** and electrical potential, $\mathbf{\Phi}$. A standard procedure of integration over the element volume is performed and the Gauss theorem is employed. An implicit time integration scheme is implemented, hence the condition that the balance equations are satisfied at the end of each time step is imposed. The system of differential equations for each finite element is:

$$\begin{bmatrix} \mathbf{M}_{uu} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \mathbf{M}_{qu} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} {}^{n+1}\ddot{\mathbf{U}} \\ {}^{n+1}\mathbf{P} \\ {}^{n+1}\ddot{\mathbf{Q}} \\ {}^{n+1}\ddot{\mathbf{\Phi}} \end{Bmatrix} + \begin{bmatrix} 0 & 0 & \mathbf{C}_{uq} & 0 \\ \mathbf{C}_{pu} & \mathbf{C}_{pp} & 0 & 0 \\ 0 & 0 & \mathbf{C}_{qq} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} {}^{n+1}\dot{\mathbf{U}} \\ {}^{n+1}\dot{\mathbf{P}} \\ {}^{n+1}\dot{\mathbf{Q}} \\ {}^{n+1}\dot{\mathbf{\Phi}} \end{Bmatrix}$$
$$+ \begin{bmatrix} \mathbf{K}_{uu} & \mathbf{K}_{up} & 0 & 0 \\ 0 & 0 & \mathbf{K}_{pq} & 0 \\ 0 & \mathbf{K}_{qp} & \mathbf{K}_{qq} & \mathbf{K}_{q\phi} \\ 0 & \mathbf{k}_{\phi p} & 0 & \mathbf{k}_{\phi\phi} \end{bmatrix} \begin{Bmatrix} \Delta\mathbf{U} \\ \Delta\mathbf{P} \\ \Delta\mathbf{Q} \\ \Delta\mathbf{\Phi} \end{Bmatrix} = \begin{Bmatrix} {}^{n+1}\mathbf{F}_u \\ {}^{n+1}\mathbf{F}_p \\ {}^{n+1}\mathbf{F}_q \\ {}^{n+1}\mathbf{F}_\phi \end{Bmatrix} \quad (4)$$

The matrices and vectors are:

$$\mathbf{K}_{q\phi} = -k_{11}^{-1}k_{12}\int_V \mathbf{N}_q^T\mathbf{N}_{\phi,x}dV \quad \mathbf{K}_{\phi q} = k_{21}\int_V \mathbf{N}_{\phi,x}^T\mathbf{N}_{q,x}dV$$
$$\mathbf{K}_{\phi\phi} = -k_{22}\int_V \mathbf{N}_{\phi,x}^T\mathbf{N}_{\phi,x}dV$$
$${}^{n+1}\mathbf{F}_q = \int_V \mathbf{N}_q^T\rho_f\,{}^{n+1}\mathbf{b}dV - \mathbf{K}_{qp}\,{}^{n}\mathbf{P} - \mathbf{K}_{qq}\,{}^{n}\mathbf{Q} - \mathbf{K}_{q\phi}\,{}^{n}\mathbf{\Phi} \quad (5)$$
$${}^{n+1}\mathbf{F}_\phi = \int_A \mathbf{N}_\phi^T\mathbf{n}^T\mathbf{j}\,dA - \mathbf{K}_{\phi p}\,{}^{n}\mathbf{P} - \mathbf{K}_{\phi\phi}\,{}^{n}\mathbf{\Phi}$$

Details about all variables in eqs (4) and (5) are given in [18]. The above equations are further assembled and the resulting FE system of equations is

integrated incrementally, with time step Δt, transforming this system into a system of algebraic equations. A Newmark integration method is implemented for the time integration.

4. Computational modeling results

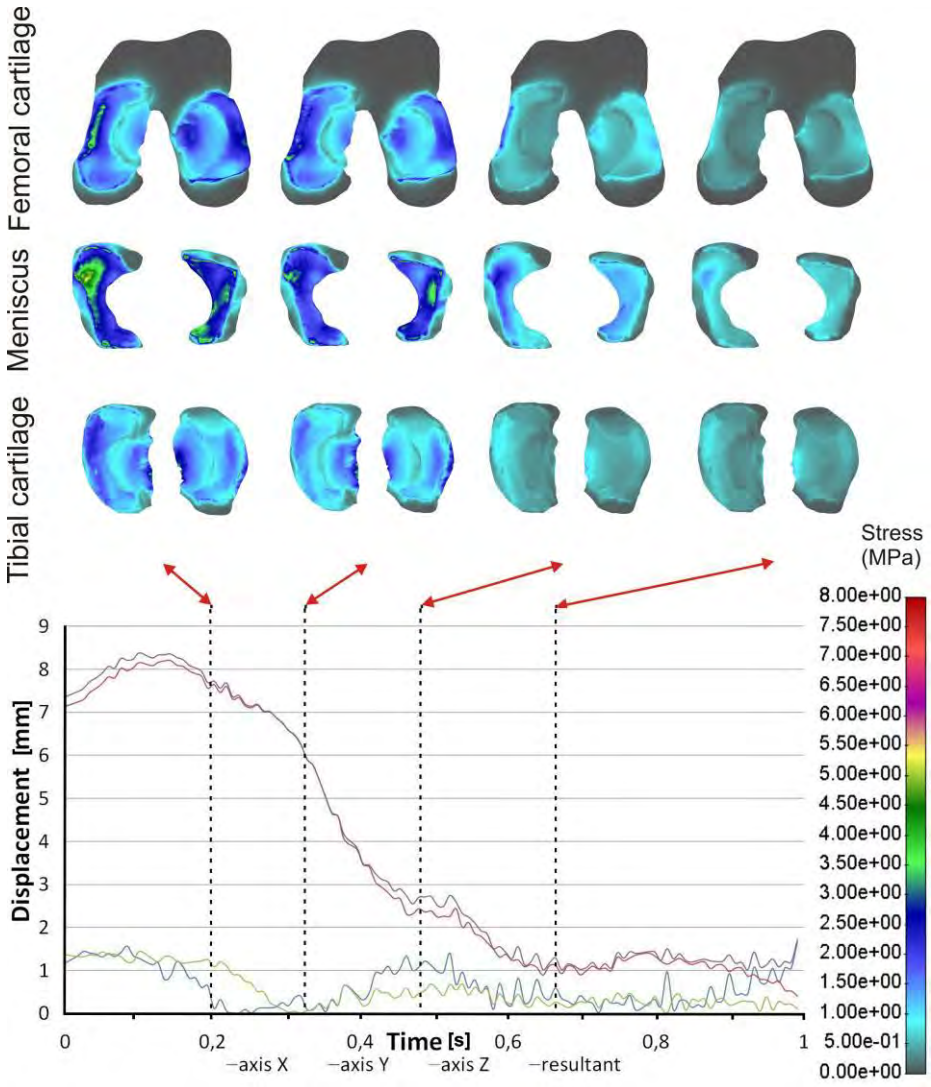


Fig. 7a. Effective von Mises stress distribution for patient specific femoral cartilage, meniscus and tibial cartilage during one gate cycle before surgery

The effective von Mises stress distribution for patient specific plane at femoral cartilage, meniscus and tibial cartilage during one gate cycle is presented in Fig. 7. It can be seen that during 30% of the gait cycle the maximum effective stress up to 8 MPa occurred and the majority of the load occurred on the meniscus part. Higher deformation of the tibia after the surgery induced higher stress on the tibial cartilage part. We presented the effective stress results for the case before (Fig 7a) and after (Fig 7b) the surgery. There is a prolonged higher stress during time cycle after surgery than before surgery.

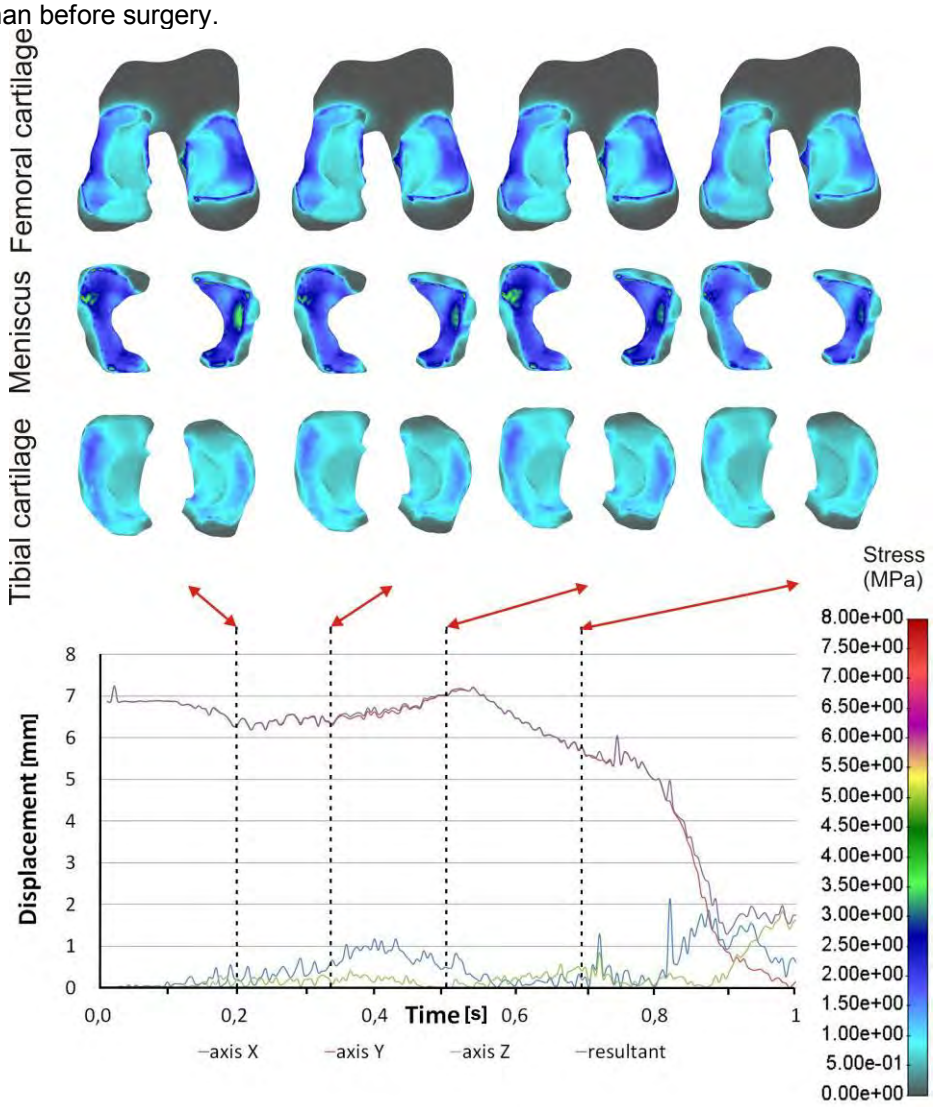


Fig. 7b. Effective von Mises stress distribution for patient specific femoral cartilage, meniscus and tibial cartilage during one gate cycle after surgery

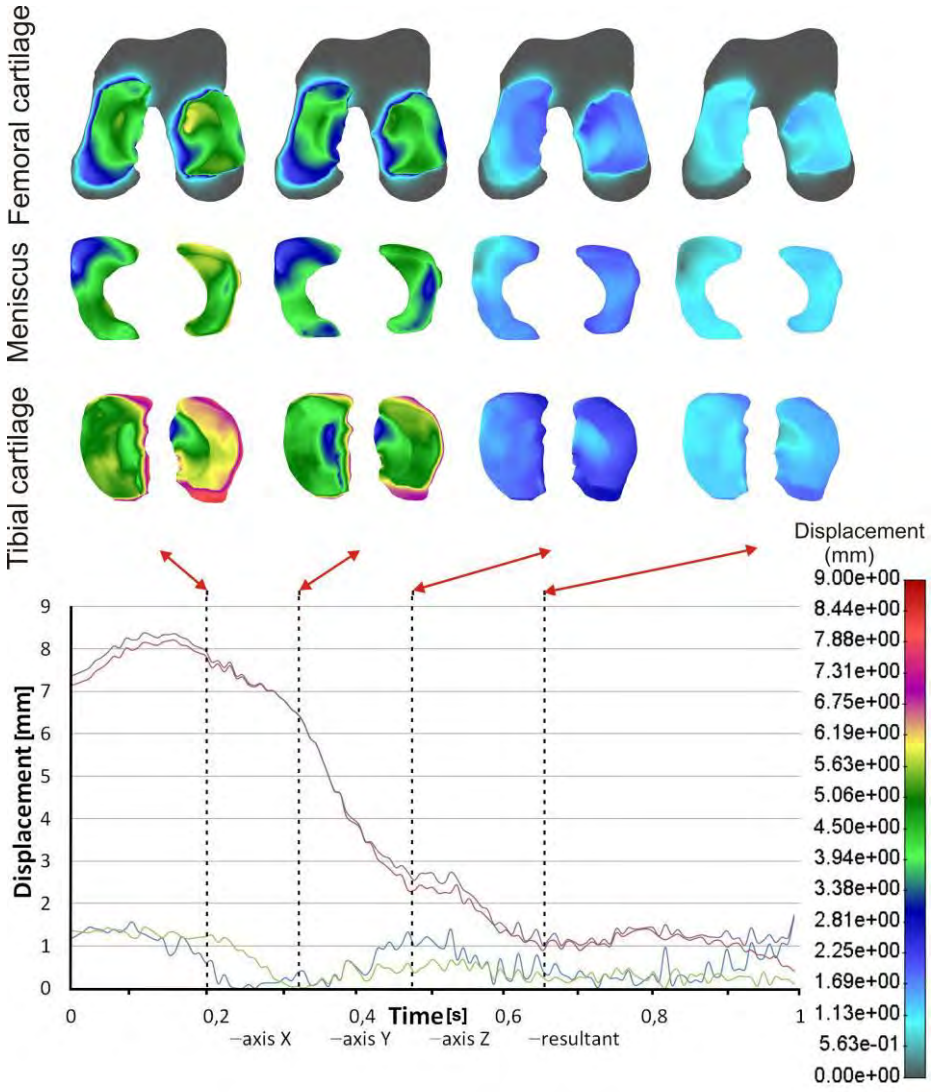


Fig. 8. Displacement distribution for patient specific femoral cartilage, meniscus and tibial cartilage during one gate cycle before surgery

The displacement distribution for patient specific femoral cartilage, meniscus and tibial cartilage during one gate cycle is presented in Fig. 8. Again it is clear that surgical intervention of ACL reconstruction establish the larger range of motion for the knee which is more stable during gate cycle analysis.

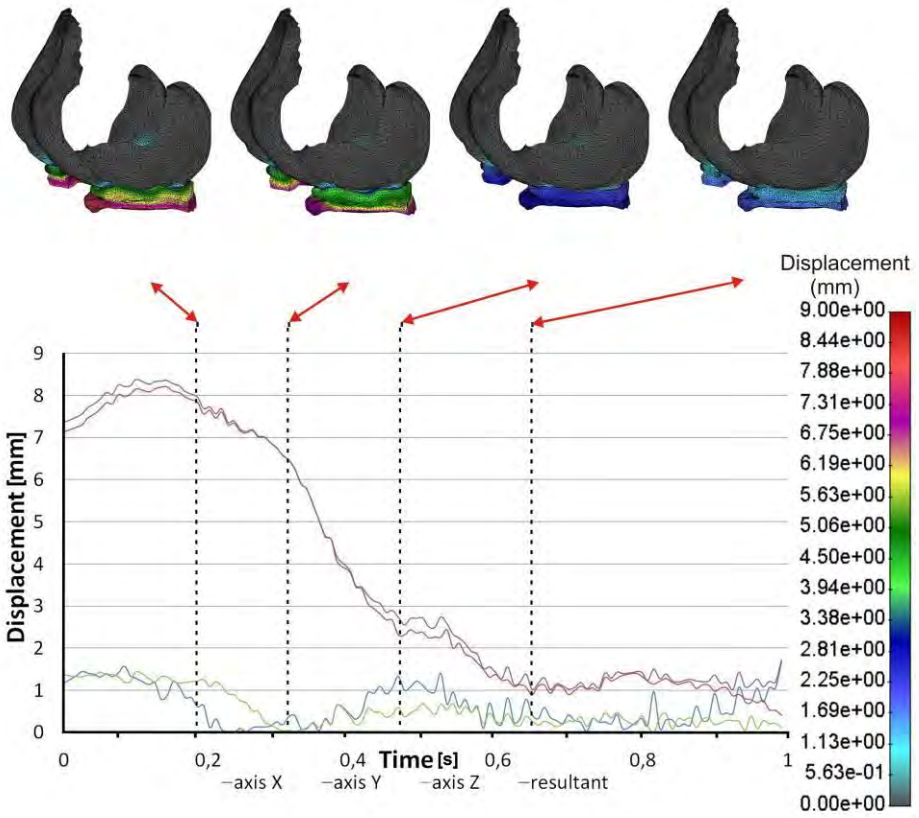


Fig. 9. Displacement distribution in the three-dimensional patient specific FE model during one gate cycle before surgery

Displacement distribution in the three-dimensional patient specific FE model during one gate cycle is presented in Fig. 9. Only femoral cartilage, meniscus and tibial cartilage are presented due to clarify. Obviously large deformation occurred in the tibial cartilage part.

Damage can occur to the tibial cartilage as an isolated condition, or in conjunction with other knee injuries. ACL injuries are commonly associated with damage to the medial and lateral surfaces of the femur and tibia. Other injuries that can lead to articular cartilage damage are those resulting from a forceful impact on the knee joint, such as a tackle in football or soccer. Injury to the articular cartilage will lead to inflammation and pain in the knee joint and in the long term it is known to accelerate the onset of osteoarthritis.

5. Discussion and conclusions

Various interventions and surgical procedures are performed for preventing knee injury. Still there is a lack of fundamental understanding of the biomechanical factors that contribute to the development and progression of knee diseases.

This study offers an innovative and robust approach to assess 3D kinetics of knee and the stress and strain distributions in the knee-based subject-specific biomechanical models of the human knee joint, MRI imaging and measured kinematic data. It could open new avenues for objective assessment of knee functioning pre and post-operation. Some details of algorithms and source code about contour recognition and 3D reconstruction are also given in this paper.

Using kinematic data measured from gait analysis we prescribe displacement on the characteristics marker position and stress and strain distributions were analyzed. It was observed that the maximum effective von Mises stress distribution up to 8 MPa was happen during 30% of the gait cycle. The location of the maximum stress occurred on the meniscus part. Increased deformation of the tibia after the surgery induced higher stress on the tibial cartilage part.

Main contribution of this study is noninvasive effective stress calculation for a specific given patient. Input data are provided from gait analysis experimental measurements and effective stress analysis is calculated from finite element analysis. This will open a new avenue for preoperative and postoperative surgical planning and treatment of the knee for specific patients.

There are also some limitations of the current study. We used material properties from literature data and it will be in future based on advanced image method for moving of the segments during MRI procedure. However, this study shows the ability of the current model to investigate the effect of different biomechanical factors on the stress at the knee joint.

Acknowledgments. This work has been partly supported by Ministry of Education and Science of Serbia, Grants No. III-41007, OI-174028 and project supported by Faculty of Medicine Kragujevac, Grant JP 20/10.

References

1. Fernandez J.W. and Pandy M. G.: Integrating modelling and experiments to assess dynamic musculoskeletal function in humans. Exp Physiol, Vol.91, No. 2, 371–382. (2006)
2. Li G, Lopez O, Rubash H.: Variability of a three dimensional finite element model constructed using magnetic resonance images of a knee for joint contact stress analysis. J Biomed Eng. Vol. 123, 341–346. (2001)

3. Li G, Suggs J, Gill T.: The effect of anterior cruciate ligament injury on knee joint function under a simulated muscle load: a three-dimensional computational simulation. Ann Biomed Eng, Vol. 30, 713–720. (2002)
4. Andriacchi TP, Briant PL, Bevill SL, Koo S.: Rotational changes at the knee after ACL injury cause cartilage thinning. Clin Orthop Relat Res, Vol. 442, 39–44. (2006)
5. Bachrach NM, Valhmu WB, Stazzone E, Ratcliffe A, Lai WM, Mow VC.: Changes in proteoglycan synthesis of chondrocytes in articular cartilage are associated with the time dependent changes in their mechanical environment. J Biomech, Vol. 28, 1561–1569. (1995)
6. Shelburne KB, Pandy MG & Torry M.: Comparison of shear forces and ligament loading in the healthy and ACL-deficient knee during gait. J Biomech, Vol. 37, 313–319. (2004)
7. Yao J, Snibbe J, Maloney M, Lerner AL.: Stresses and strains in the medial meniscus of an acl deficient knee under anterior loading: a finite element analysis with image-based experimental validation. J Biomech Eng. Vol. 128, 135–141. (2006)
8. Yang N.H., Canavan P.K., Nayeb-Hashemi H., Najafi C., Vaziri A.: Protocol for constructing subject - specific biomechanical models of knee joint. Computer Methods in Biomechanics and Biomedical Engineering, Vol. 13, 589 - 603. (2010)
9. Matić A., Ristić B., Devedžić G., Filipović N., Petrović S., Mijailović N., Ćuković S.: Gait analysis in patients with chronic anterior cruciate ligament injury, Serbian Journal of Experimental and Clinical Research, Vol.13, No. 2, 49 - 54. (2012)
10. Brandsson S., Karlsson J., Swärd L., Kartus J., Eriksson B.I., Kärrholm J: Kinematics and Laxity of the Knee Joint After Anterior Cruciate Ligament Reconstruction. The American Journal of Sports Medicine, Vol. 30, 361 – 367. (2002)
11. Isaac D.I., Beard D.J., Price A.J., Rees J., Murray D.W., Dodd C.A.F.: In-vivo sagittal plane knee kinematics: ACL intact, deficient and reconstruction knees. The Knee, Vol. 12, 25 - 31. (2005)
12. Manal K., McClay Davis I., Galinat B., Stanhope S: The accuracy of estimating proximal tibial translation during natural cadence walking: bone vs. skin mounted targets. Clinical Biomechanics Vol. 18, 126 – 131. (2003)
13. Gao B., Cordova M.L., Zheng N.N.: Three-dimensional joint kinematics of ACL-deficient and ACL-reconstructed knees during stair ascent and descent. Human Movement Science, Vol. 31, 222 -235. (2012)
14. Kvist J.: Tibial translation in exercises used early in rehabilitation after anterior cruciate ligament reconstruction Excercises to achieve weight-bearing. The Knee, Vol. 13, 460 - 463. (2006)
15. Scanlan S.F., Chaudhari A.M.W., Dyrby C.O., Andriacchi T.P.: Differences in tibial rotation during walking in ACL reconstructed and healthy contralateral knees. Journal of Biomechanics, Vol. 42, 1817 – 1822. (2010)
16. Freitag L, Plassmann P.: Local optimization based simplicial mesh untangling and improvement. Intl. J. Num. Method. in Engr., Vol. 49, 109-125. (2000)
17. Tsuda, A., Filipovic, N., Haberthür, D., Dickie, R., Matsui, Y., Stampanoni, M. and Schittny, J.C.: Finite element 3D reconstruction of the pulmonary acinus imaged by synchrotron X-ray tomography. J Appl Physiol, Vol. 105, 964-976. (2008)
18. Filipovic N., Basics of Bioengineering. Monograph in Serbian. (2012)

Appendix:

```
void CContourRecognizer::LoadPng(const char *pName)
{
	WCHAR wbuf[512];
	mbstowcs(wbuf, pName, sizeof(wbuf)/sizeof(wbuf[0]));
	Bitmap bmp(wbuf);
	int w = bmp.GetWidth();
	int h = bmp.GetHeight();
	m_Pixmap.InitDim(w, h);
	for(int x=0;x<w;x++)
	{
		for(int y=0;y<h;y++)
		{
			Color cl;
			bmp.GetPixel(x,h-y-1,&cl);
			double tr = cl.GetR()/255.0;
			//if (tr > 0.9) tr = 0;
			m_Pixmap.SetAt(x,y, tr);
		}
	}
}

void CContourRecognizer::LoadDicomFile(const char *pName)
{
	m_Pixmap.Kill();
	CDCMFile dcm;
	dcm.Read(pName);
	int w = dcm.m_nWidth;
	int h = dcm.m_nHeight;
	m_Pixmap.InitDim(w, h);
	for(int x=0;x<w;x++)
	{
		for(int y=0;y<h;y++)
		{
			m_Pixmap.SetAt(x,y, dcm.GetPixelValueD(x,h-y-1));
		}
	}
}

bool CContourRecognizer::Mesh_Generation(const Math3d::M2d &InnerPoint, const Math3d::M2d &OutterPoint)
{
	const int pixCount = 2048;
	const double dInvStep = 1.0 / (double)pixCount;
	double pixValues[pixCount];
	for(int i=0;i<pixCount;i++)
```

```
        {
                double t = i * dInvStep;
                Math3d::M2d cur = InnerPoint * (1-t) + OutterPoint * t;
                pixValues[i] = m_Pixmap.GetLinearWSlopesAt(cur.x, cur.y);
        }
        double tMid;
        for(i=1;i<pixCount;i++)
        {
                double d0 = pixValues[i-1];
                double d1 = pixValues[i];
                if ((d0-m_dLevel)*(d1-m_dLevel) <= 0)
                {
                        tMid = (i-0.5)*dInvStep;
                        break;
                }
        }
        if (i == pixCount) return false;
        return FindIsoPoints(InnerPoint * (1-tMid) + OutterPoint * tMid);
  }

  bool CContourRecognizer::Surface_Generation(const Math3d::M2d
&firstPoint)
  {
        m_IsoPoints.RemoveAll();
        m_IsoPoints.SetSize(0,2000);
        double dLevel = m_Pixmap.GetLinearWSlopesAt(firstPoint.x,
firstPoint.y);
        Math3d::M2d curPoint = firstPoint;
        double curLevel = dLevel;
        Math3d::M2d grad;
        double dStep = m_dStep;
        double dStepSqr = dStep*dStep*1.01;
        int nMaxPts = 3000;
        for(int i=0;i<nMaxPts;i++)
        {
                grad.x = m_Pixmap.GetLinearDifX(curPoint.x, curPoint.y,
0.001);
                grad.y = m_Pixmap.GetLinearDifY(curPoint.x, curPoint.y,
0.001);
                Math3d::M2d moveVec = grad;
                moveVec.Rotate90();
                moveVec.Normalize();
                curPoint += moveVec * dStep;
                for(int c=0;c<3;c++)
                {
                        grad.x = m_Pixmap.GetLinearDifX(curPoint.x,
curPoint.y, 0.0001);
```

```
                        grad.y = m_Pixmap.GetLinearDifY(curPoint.x,
curPoint.y, 0.0001);
                        curLevel =
m_Pixmap.GetLinearWSlopesAt(curPoint.x, curPoint.y);
                        Math3d::M2d corr = grad * ((dLevel - curLevel) /
grad.NormSqr());
                        curPoint += corr;
                }
                m_IsoPoints.Add(curPoint);
                if (m_IsoPoints.GetSize() > 5)
                {
                        if ((curPoint-firstPoint).NormSqr() <= dStepSqr)
break;
                }
        }
        if (i == nMaxPts) ::AfxMessageBox("Can not close contour");
        return (i != nMaxPts);
  }
```

Nenad Filipović received the Ph.D. in bioengineering from the University of Kragujevac, Serbia in 1999. He was Research Associate at Harvard School of Public Health in Boston, USA. He is currently a Professor in Bioengineering at Faculty of Mechanical Engineering, University of Kragujevac, Serbia. His research interests are in the area of fluid mechanics, coupled problems; fluid-structure interaction, heat transfer; biofluid mechanics; biomechanics, multi-scale modeling, discrete modeling, molecular dynamics, computational chemistry and bioprocess modeling. He is author and co-author 6 textbooks and 1 monograph on English language, over 50 publications in peer review journals and over 5 software for modeling with finite element method and discrete methods from fluid mechanics and multiphysics. He leads a number of national and international projects in area of bioengineering.

Velibor Isailović has PhD in area of bioengineering at Metropoliten University, Serbia. His main research interests include coupled problem, fluid-structure interaction, particle methods, finite element methods. He has authored/co-authored more than 8 papers in peer-review journals.

Dalibor Nikolić is PhD student at Faculty of Engineering in University of Kragujevac. His research interests include computer graphics, medical imaging reconstruction, finite element methods and software engineering. He is main research software engineer at Center for Bioengineering at Faculty of Engineering.

Aleksandar Peulić received the Diploma degree in electronic engineering from Faculty of Electronic Engineering, University of Nis, Nis, Serbia, in 1994, the Master of Science in electrical engineering from Faculty of Electronic Engineering, University of Nis, Nis, Serbia, in 2000 and the Ph.D. degree in electrical engineering from the University of Kragujevac, Serbia, in 2007. From avg. 2008 to feb. 2009, he had a postdoctoral education at the University of Alabama in Huntsville. His research interests include microcontrollers systems, wearable sensors and bioengineering. He is Assistant Professor at Technical faculty, University of Kragujevac.

Nikola Mijailović is PhD Student at Faculty of Engineering in University of Kragujevac. His research interests include electronic and hardware development for bioengineering devices and systems. He has also interest in medical imaging and neural network.

Suzana Petrović is R&T assistant at Faculty of Engineering, University of Kragujevac, Serbia. Her major research interests are bioengineering, gait analysis, machine learning, statistics and advanced product and process development. She is coauthor of one book on 3D product modeling and coauthor of chapter in book on CAD/CAM technology. She is also the author or co-author of more than 10 papers in international and national journals or papers presented at international and national conferences.

Saša Cuković is teaching and research assistant and PhD candidate at Faculty of Engineering, University of Kragujevac, Serbia. He was scholarship holder of Ministry of Education, Science and Technological Development of Republic of Serbia and DAAD grant holder at Technical University of Munich. His main research interests include CAD/CAMsystems, reverse engineering and noninvasive 3D reconstruction and modeling in engineering and medicine, augmented reality and computer vision. He has authored/co-authored more than 20 papers.

Radun Vulović is PhD student at Faculty of Engineering in University of Kragujevac. His research interests include sports biomechanics, computer graphicsfinite element methods and software engineering. He is research software engineer at Center for Bioengineering at Faculty of Engineering.

Aleksandar Matić is othopaedic surgeon at Clinical Center Kragujevac, and T&R assistant at the Faculty of Medical Sciences at University Kragujevac (Serbia). His main research interests involve sports injuries and gait analysis. He has authored/coauthored more than 30 research papers, published in national journals or presented at international and national conferences.

Nebojša Zdravković received the Ph.D. in bioengineering from the University of Kragujevac, Serbia in 2000. He is currently a Associate Professor in Informatics and Medical Statistics at Faculty of Medical Science, University of Kragujevac, Serbia His research interests are in data mining,

statistical methods, finite element method fluid mechanics, coupled problems.

Goran Devedžić is Professor at Faculty of Engineering, University of Kragujevac, Serbia. His research interests focus on the advanced product and process development, industrial and medical application of soft computing techniques, and bioengineering. He has authored/co-authored more than 100 research papers, published in international and national journals or presented at international and national conferences, as well as three books on CAD/CAM technology and 3D product modeling.

Branko Ristić is Associate Professor at Faculty of Medical Sciences, University of Kragujevac, Serbia. His research interests include hip and knee arthroplasty and bioengineering of musculo-skeletal system. He is the author/co-author of more than 100 research papers, six chapters in books of orthopedics and bioengineering. He was actively involved in several research projects addressing bioengineering and total hip and total knee arthroplasty. Currently he is a Head of Clinic for Orthopedics and Traumatology at Clinical Center Kragujevac, Serbia, and the President of the Regional Medical Chamber for Central and Western Serbia.

Received: May 31, 2012; Accepted: December 31, 2012.

DOI:10.2298/CSIS120531015P

Modeling of Arterial Stiffness using Variations of Pulse Transit Time

Aleksandar Peulić[1], Natasa Milojević[2], Emil Jovanov[3], Miloš Radović[4], Igor Saveljić[4], Nebojša Zdravković[5], Nenad Filipović[4]

[1] Technical Faculty, University of Kragujevac, Svetog Save 65, 32000 Cacak, Serbia
[2] Intellectual Property Office, Kneginje Ljubice 5, 11000 Belgrade, Serbia
[3] Electrical and Computer Engineering, University of Alabama, Huntsville, AL 35899 USA
[4] Faculty of Engineering, University of Kragujevac, Jovana Cvijica bb, 34000 Kragujevac, Serbia
[5] Medical Faculty, University of Kragujevac, Svetozara Markovica 69, 34000 Kragujevac, Serbia

Abstract. In this paper, a finite element (FE) modeling is used to model effects of the arterial stiffness on the different signal patterns of the pulse transit time (PTT). Several different breathing patterns of the three subjects are measured with PTT signal and corresponding finite element model of the straight elastic artery is applied. The computational fluid-structure model provides arterial elastic behavior and fitting procedure was applied in order to estimate Young's module of stiffness of the artery. It was found that approximately same elastic Young's module can be fitted for specific subject with different breathing patterns which validate this methodology for possible noninvasive determination of the arterial stiffness.

Keywords: arterial stiffness, finite element modeling, microcontroller, pulse transit time.

1. Introduction

Pulse Transit Time (PTT) represents the time which blood pulse wave propagates from heart to a peripheral artery measurement site (for example, the finger). It has an important role in noninvasive assessment of blood pressure. The components of photoplethismogram (PPG) signal are not fully understood. It is generally accepted that they can provide valuable information about the cardiovascular system [1]. PTT is measured using

electrocardiogram (ECG) and (PPG) [2]. The pulse pressure waveform is getting from the left ventricle blood ejection into the aorta. There are two main parameters of the wave pulse: the blood velocity and pulse wave velocity. The blood velocity at the aorta is several meters per second and it slows down to several mm/s in the peripheral network while the pressure pulse travels much faster than blood [3]. Pulse wave velocity (PWV) is a direct measurement of arterial stiffness and describes how quickly a blood pressure pulse travels from one point to another in the human body and the time spent for such process is the PTT [4]. The value of the PWV, is affected by several factors, such as the elasticity of arterial wall, arterial geometry (radius and thickness), and blood density. Typical value ranges from 5 m/s to 15 m/s, depending on the age of people and the state of their arteries [5].

2. Related work

Atherosclerosis is a disease that increases the thickness and rigidity of the arterial blood vessels, causing narrowing of artery. The increased inflexibility of the arterial wall increases PWV, since the energy of the blood pressure pulse cannot be stored in an inflexible wall. PWV can be used as a predictor of cardiovascular mortality in hypertensive subjects [6].

In a number of other studies [7], [8], [9], [10], when comparing the results of the analysis of heart rate variability and pulse rate variability, some differences are revealed. That can be explained by temporal changes of hemodynamic parameters of the vascular system controlled by vascular regulation. Some authors tried to find characteristics defined in the frequency domain and the characteristics of the vascular system in order to obtain a diagnostic index of the vascular system, which may be used to estimate arterial stiffness [9], [10].

In this paper the finite element modeling of cardiovascular pulsation with combination of the PTT measurements on the finger of left hand is described, and we present experimental measurements, computational strategy and results of modeling validated by experiments. The main aim of this study was to find correlation of the arterial elastic Young's module with PTT.

3. Experimental measurements

Pulse travel time is measured as time period between contraction of the ventricles and arrival of the blood pulse to the left index finger. Contraction of the ventricles is detected using R peak of the ECG signal; arrival of the blood pulse is detected as the half rise point of PPG signal. Electrodes are connected with wires to the recording device which is sometimes connected with wires to the processing device. This configuration is not convenient to the patient and to the researches too. The wires make some movements

difficult and they alter normal behavior of a patient. Taking this into consideration we selected elastic belt with embedded electrodes and capturing device on the belt. The capturing device sends ECG R peak signal wirelessly to the processing board. PPG sensor is attached to the finger or ear and connected to the processing board. Processing board captures ECG signal and PPG signal and determines delay between consecutive R peaks for ECG and PPG signals. For further processing it should have sufficient amount of memory, low frequency clock and timer with capture/compare registers. Processing board can display result on LCD and/or transmit the result to PC via Ethernet or serial line. The overall hardware architecture is shown in Fig. 1 and PTT timing diagram in Fig. 2.

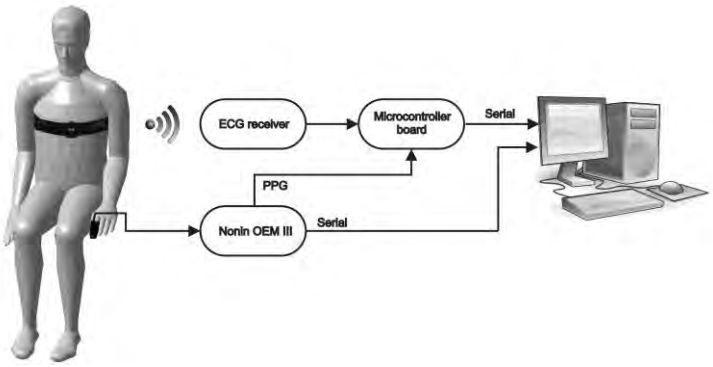


Fig. 1. Hardware architecture

Precise measurement of the PTT with minimum latency is implemented using digital output of the PPG sensor (OEM III board from Nonin [11]) and custom ECG peak capture board connected to the input pins of timer on the processing board. Timer capture/compare registers used input pin interrupts to capture relative timing of ECG and PPG signals. Measured intervals are sent to PC for display and processing. Fixed point FIR filtering routines for ECG signal are implemented in firmware, code below:

```
  int filterlp(int sample)
// Lowpass FIR filter for EKG
  {   static int buflp[32];
// Reserve 32 loactions for circular buffering
      static int offsetlp = 0;
      long z;
      int i;
      buflp[offsetlp] = sample;
      z = mul16(coeffslp[8], buflp[(offsetlp - 8) &
0x1F]);
      for (i = 0;  i < 8;  i++)
```

```
      z += mul16(coeffslp[i], buflp[(offsetlp - i) &
0x1F] + buflp[(offsetlp - 16 + i) & 0x1F]);
      offsetlp = (offsetlp + 1) & 0x1F;
      return  z >> 15;
// Return filter output
  }//

  int filterhp(int samplehp)
  // Highpass FIR filter for hear rate
  {   static int bufhp[32];
  // Reserve 32 loactions for circular buffering
      static int offsethp = 0;
      long z;
      int i;
      bufhp[offsethp] = samplehp;
      z = mul16(coeffshp[8], bufhp[(offsethp - 8) &
0x1F]);
      for (i = 0;  i < 8;  i++)
      z += mul16(coeffshp[i], bufhp[(offsethp - i) &
0x1F] + bufhp[(offsethp - 16 + i) & 0x1F]);
      offsethp = (offsethp + 1) & 0x1F;
      return  z >> 15;
// Return filter output
  }//
```

Code below represents main firmware procedure and sends the two bytes of the ADC register to the PC Application via USB interface:

```
  void main(void)
  {   Init ();
// Initialize device for the application
      while(1)
     {LPM0;
  // Enter LPM0 needed for UART TX completion

      Dataout = filterlp(Datain);
  // Lowpass FIR filter for filtering out 60Hz
      Dataout_pulse = filterhp(Dataout)-128;
  // Highpass FIR filter to filter muscle artifacts
      Dataout = Dataout >> 6;
  // Scale Dataout to use PC program

      if(Dataout>255)
// Set boundary 255 max
        Dataout=255;
//
      if(Dataout<0)
// Set boundary 0 min
        Dataout=0;
//
```

```
      samples[0] = Dataout;
      send_data();
//sends

      counter++;
// Debounce counter
      pulseperiod++;
// Pulse period counter
      if (Dataout_pulse > 48)
// Check if above threshold
      { LCDM10 |= 0x0f;
// Heart beat detected enable "^" on LCD
      counter = 0;}
// Reset debounce counter
      if (counter == 128)
// Allow 128 sample debounce time
      {LCDM10 = 0x00;
// Disable "^" on LCD for blinking effect
      beats++;
      if (beats == 3)
      {beats = 0;
      heartrate = itobcd(92160/pulseperiod);
  // Calculate 3 beat average heart rate per min
      pulseperiod = 0;
  // Reset pulse period for next measurement
      LCDMEM[0] = char_gen[heartrate & 0x0f];
  // Display current heart rate units
      LCDMEM[1] = char_gen[(heartrate & 0xf0) >> 4];
// tens
      LCDMEM[2] = char_gen[(heartrate & 0xf00) >> 8];}}
// hundreds
      }
   }
  }//main
  void send_data(void)
  {
      for (j=0;j<1;j++)
    {
     while (!(IFG1 & UTXIFG0));
  // USART0 TX buffer ready?
     TXBUF0 = samples[j];
    }

  }
```

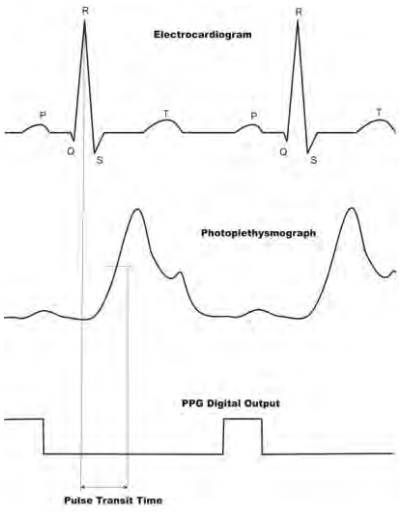


Fig. 2. PTT illustration. RR is time interval between two R peaks in the ECG

In order to validate modeling of arteries in different physiological conditions, eight breathing experiments were conducted. The goal of these experiments was to record difference in RR (RR is time interval between two R peaks in the ECG signal) and PTT measurements due to different breathing patterns and to model the changes using Finite Element (FE) modeling. The experiment was conducted in the sitting position and consisted of few steps:

Subject_1:

normal breathing: 1 minute; paced breathing at 5sec/breath for two minutes;

paced breathing at 10 sec/breath for two minutes;normal breathing 1 minute

Subject_2:

normal breathing: 5 minute;paced breathing at 5sec/breath for five minutes;

Subject_3:

normal breathing: 5 minute;paced breathing at 5sec/breath for five minutes.

PTT measurements for all subject are presented in Fig. 3 and RR measurements are given in Fig. 4.

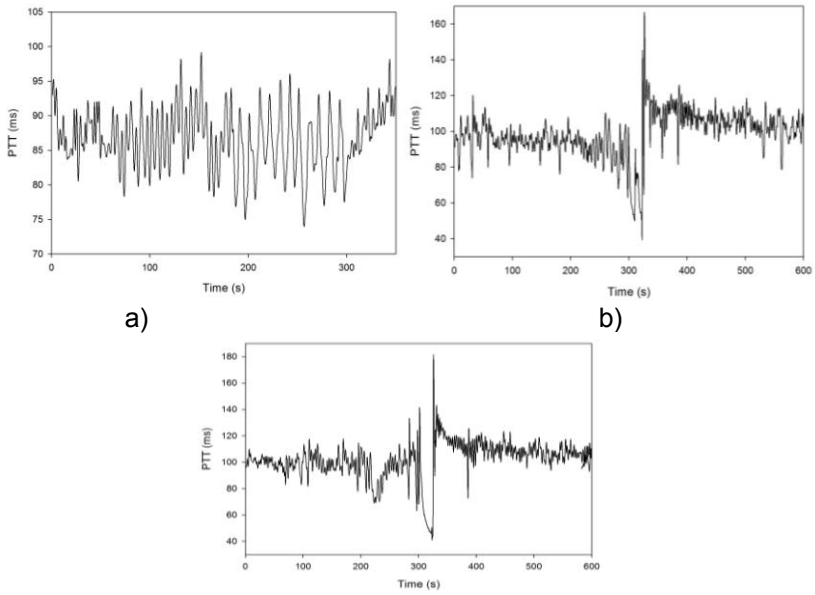


Fig. 3. PTT measurements: a) Subject_1, b) Subject_2, c) Subject_3

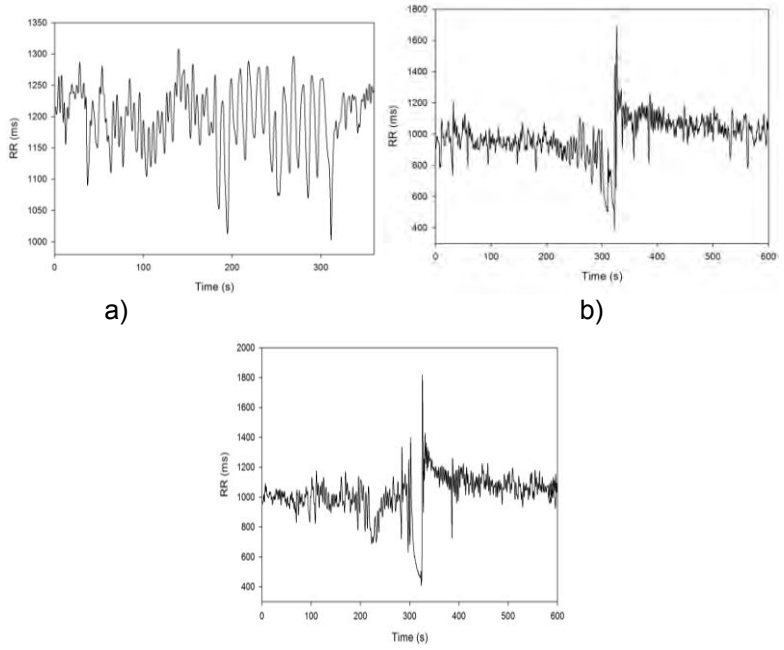


Fig. 4. RR measurements: a) Subject_1, b) Subject_2, c) Subject_3

4. Computational modeling results

Computer modeling is a useful tool for the study of dynamic behavior of blood flows in the arteries.

PWV provides an assessment of the state of the cardiovascular system. Pressure pulse velocity is determined by elastic and geometric properties of arterial wall. It can be expressed by the Bramwell-Hill equation [2]

$$PWV = \sqrt{\frac{V}{\rho}\frac{\Delta P}{\Delta V}} \tag{1}$$

where V is blood volume, ΔV and Δp are the changes of blood volume and pressure, and ρ is the density (of blood). This relation enables the study of compliance (arterial elasticity) by measuring the PWV. Also, Moens-Korteweg equation can be used to determine a PWV [2]

$$PWV = \sqrt{\frac{\Delta E h_w}{2\rho_w r_i}} \tag{2}$$

where ΔE is the incremental elastic modulus, ρ_w is wall density, h_w is the thickness of the arterial wall and r_i is the internal vessel radius. The Bramwell-Hill Eq. (1) and the Moens-Kortweg Eq. (2) are the same equations, just with different denotation. Eq (1) is more applicable in medical technology because gives direction relation of quantities ΔV and Δp. Actually a small rise in pressure may be shown to cause a small increase, in the radius of the artery, or a small increase, in its own volume *V* per unit length. Both Eqs. (1) and (2) can be derived from Newton's equation for wave speed using the substitution of the equation of the bulk modulus in terms of volumetric strain. Relationship between PTT and PWV can be represented as [12]

$$PTT = \frac{L}{PWV} \tag{3}$$

where L is the distance the pulse travels (roughly equals to the aorta ascending arch plus arm length). In this paper, we simulate a blood vessel as an elastic tube where fluid flows in a periodic time function, which corresponds to the periodic heart rate. We model the straight artery of length 700 mm, lumen radius 3.8 mm and wall thickness 0.5 mm. The comparison of analytical solution from Moens-Korteweg theory (2) and numerical solutions by varying Young's modulus is shown in Fig. 5. A very good agreement of numerical solutions with analytical results is achieved, where mean difference of the two curves is 0.10 m/s, standard deviation has a value of 0.0478 m/s, while the standard error is 0.0169.

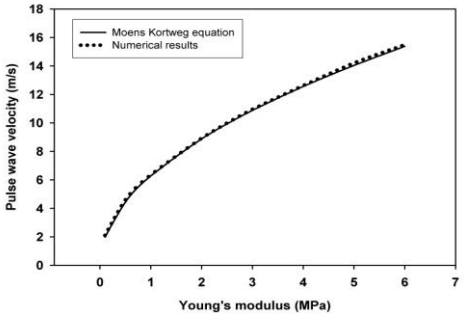


Fig. 5. Pulse wave velocity as function of incremental Young's modulus. Comparison of numerical and analytical solution

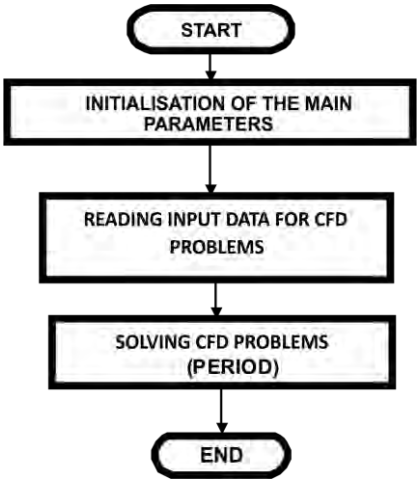


Fig. 6 Flow chart for program for finite element method for CFD

Global FLOW CHART for Program for Finite element method is presented in Fig 6. After initialization of the main parameters of the program, input data as finite element nodes and elements, boundary conditions, initial conditions, material properties, time step function are reading. Then finite element solver is running for CFD problems. The solver is un-symmetric because pressure velocity formulation. For faster calculation we implemented a penalty formulation in our solver [14], [15].

The incremental-iterative form of the equations for time step Δt and equilibrium iteration "*i*" are:

$$\begin{bmatrix} \frac{1}{\Delta t}\mathbf{M}_{\mathbf{v}} + {}^{t+\Delta t}\mathbf{K}_{\mathbf{vv}}^{(i-1)} + {}^{t+\Delta t}\mathbf{K}_{\mu\mathbf{v}}^{(i-1)} + {}^{t+\Delta t}\mathbf{J}_{\mathbf{vv}}^{(i-1)} & \mathbf{K}_{\mathbf{vp}} \\ \mathbf{K}_{\mathbf{vp}}^{\mathbf{T}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \Delta\mathbf{v}^{(i)} \\ \Delta\mathbf{p}^{(i)} \end{Bmatrix} = \begin{Bmatrix} {}^{t+\Delta t}\mathbf{F}_{\mathbf{v}}^{(i-1)} \\ {}^{t+\Delta t}\mathbf{F}_{\mathbf{p}}^{(i-1)} \end{Bmatrix} \quad (4)$$

The left upper index "t+Δt" denotes that the quantities are evaluated at the end of time step. The matrix M_v is a mass matrix, K_{vv} and J_{vv} are convective matrices, $K_{\mu v}$ is the viscous matrix, K_{vp} is the pressure matrix, and F_v and F_p are forcing vectors. The pressure is eliminated at the element level through the static condensation. A parallel version of the solver is used.
In addition to the velocity field, the wall shear stress computation is performed. The mean shear stress τ_{mean} within a time interval T is calculated as [14]

$$\tau_{mean} = \left| \frac{1}{T} \int_0^T t_s dt \right| \quad (5)$$

where $\boldsymbol{t}_s$ is the surface traction vector. Another scalar quantity is a time-averaged magnitude of the surface traction vector, calculated as

$$\tau_{mag} = \frac{1}{T} \int_0^T |t_s| dt \quad (6)$$

The interface for fluid-structure interaction algorithm is presented in Fig. 7.

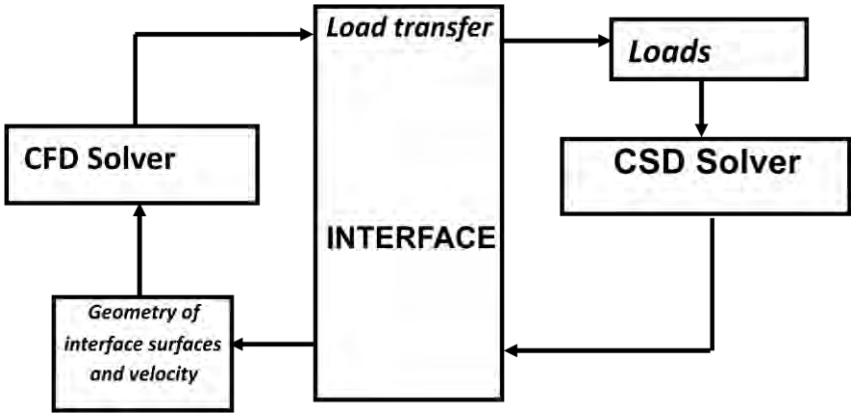


Fig. 7. Information exchange for the coupled problem fluid-structure interaction

We use the loose coupling solution algorithm, ([16]) for the fluid-structure interaction problem. The loose coupling has many advantages with respect to

the strong coupling approach. The main goal is to reuse CFD and CSD codes with a minimum of modifications. The fluid and solid variables are updated alternatively by the independent CFD and CSD codes with exchange boundary information at each time step, as it is shown in Fig. 7.

The CFD solver needs the position and the velocity of the interface surface. This information is a part of the solution of the CSD problem. On the other hand, the CSD solver takes the fluid loads on the interface surface from the CFD solver as the boundary conditions and solves for the deformation. The most attractive feature of the loose coupling approach is that the CFD and CSD solvers do not need to be rewritten.

In the initialization phase the master code first reads the necessary input file for the CSD and CFD solvers, initializes the arrays, and identifies the points common for the fluid and solid domains. These points are the ones used for the exchange information between CFD and CSD codes.

The master code drives the solution of the coupled problem. It contains a global loop in which the fluid and structure solvers are called alternatively. Inside the global loop several major operations are: call the CFD solver to advance the fluid solution, transfer the computed loads to the solid, call the CSD solver to update the solid solution, and transfer the interface position to the fluid. The global algorithm of the procedure is shown in Fig. 8 [16].

The finite element mesh discretization of the straight blood vessel is shown in Fig. 9a. For fluid domain we used 3D linear 8-node finite element method with 8 linear velocity shape function and constant pressure per element. Penalty method was implemented in order to reduce time for calculation. For solid domain 3D linear 8-node finite element is used with 8 linear displacement shape function. Linear material model and geometry linear analysis with small deformation is used. The fluid domain is discretized with structured mesh of 7400 fluid nodes and solid domain contains 5600 solid nodes. Blood is taken as an incompressible Newtonian fluid, which is appropriated for the large arteries. The blood density is $\rho=1.025$ g/cm^3, and the kinematics viscosity is $v=0.035$ cm^2/s [13].

The contour slice of velocity magnitude and wall shear stress are shown in Figs. 10 and 11 for early flow deceleration t=0.4 s, in the case of deformable and rigid walls.

The Fig. 10 shows the difference between the velocity of fluid in case of deformable and rigid walls. Deformable walls show lower fluid velocity because greater radial displacement which reduces the speed of the fluid. Distributions of wall shear stress for early flow deceleration (t=0.4 s) are shown in Fig. 11.

The results for wall shear stress (Fig. 11) show generally lower shear stresses for deformable walls in comparison to the rigid wall. This is due to more continuous changes of the velocities near the walls (smaller velocity gradient) under pulsatility conditions when the walls are considered as deformable media.

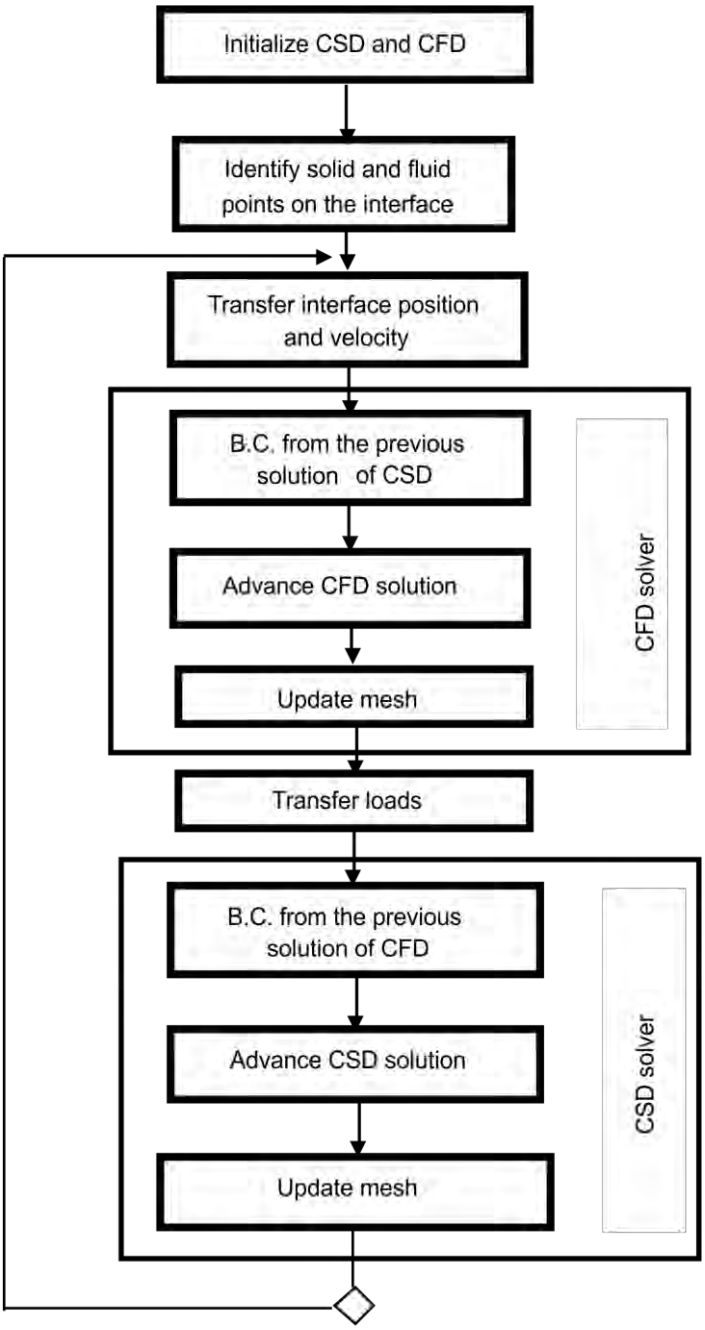
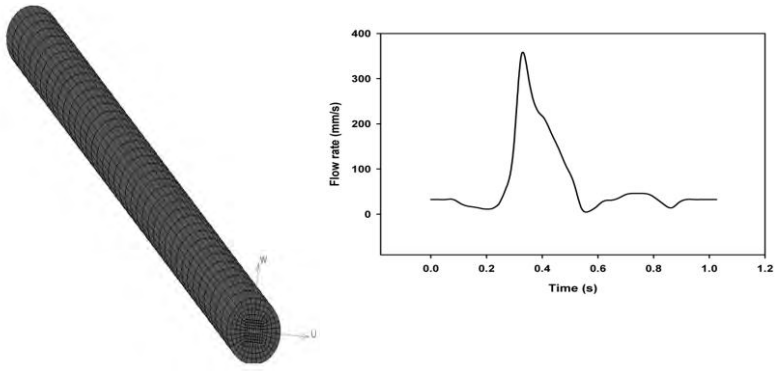
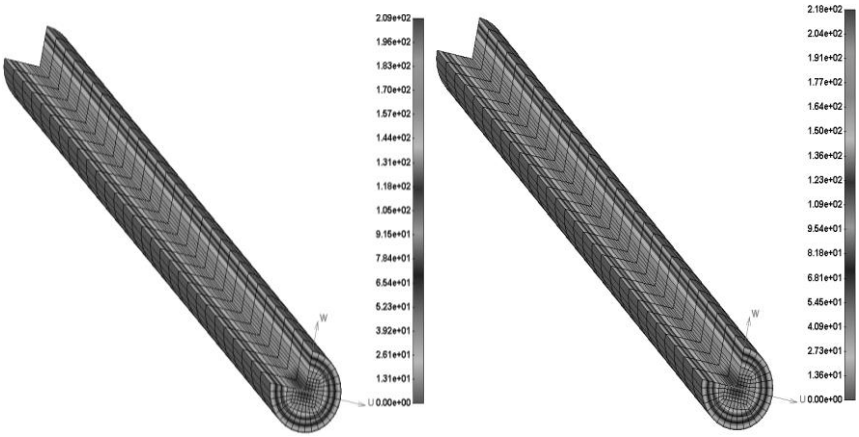


Fig 8. Algorithm for solving fluid-structure problem



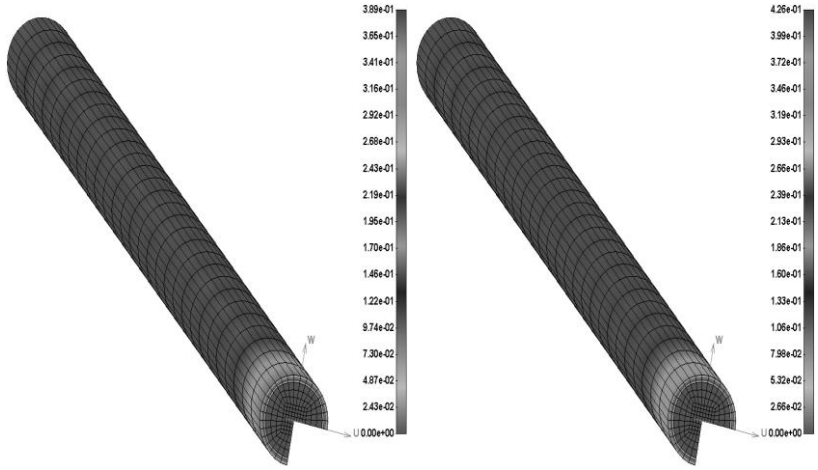
a) b)

Fig. 9. FE model of the blood vessel: a) Finite element mesh; b) Input flow rate vs. time (pulsatile flow)



a) b)

Fig. 10. The velocity magnitude field in the blood vessel for early deceleration flow t=0.4 s: a) Deformable walls; b) Rigid walls



a) b)

Fig. 11. The wall shear stress in the blood vessel model for early flow deceleration t=0.4 s: a) Deformable walls; b) Rigid walls

A great attention was devoted to matching the results obtained from our models with the results obtained by measurements described in the section experimental measurements. In fact, we tried to fit Young's modulus for all different physiological states (experimental breathing protocols), which represent the relationship between PTT signal and pressure.

To fit elasticity modules (for each breathing pattern) we used a simplex optimization method developed by John Nelder and Roger Mead [17]. This method is extremely simple and involves only function evaluations (no derivatives). Function that was minimized is calculated as a sum of eight squared error functions - one for each breathing pattern. These squared error functions represent the difference between measured PTT values and PTT values calculated by substituting modules E in (2).

Each pattern of breathing is simulated by time functions of the flow of fluids. As a result of these input functions, we obtained the corresponding pulse wave velocities that are inserted in PTT formula (3) and get a set of values for the PTT signal. Our task was to fit the module E, so that the calculated PTT signal corresponds to the measured human values. The fitting results are shown in Fig. 12, 13 and 14.

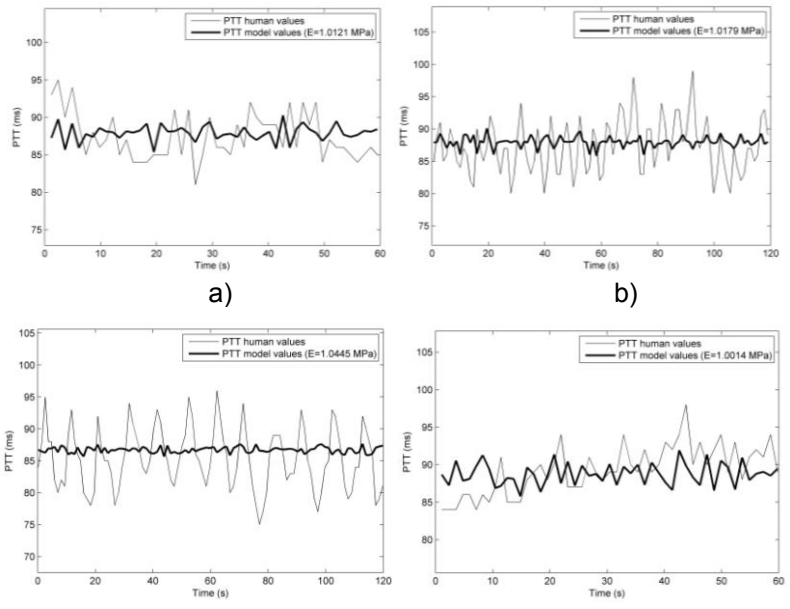


Fig. 12. The fitting results of modules E_1, E_2, E_3 and E_4 for Subject_1: a) Fitting result of first pattern breathing; b) Fitting result of second pattern breathing; c) Fitting result of third pattern breathing, d) Fitting result of fourth pattern breathing

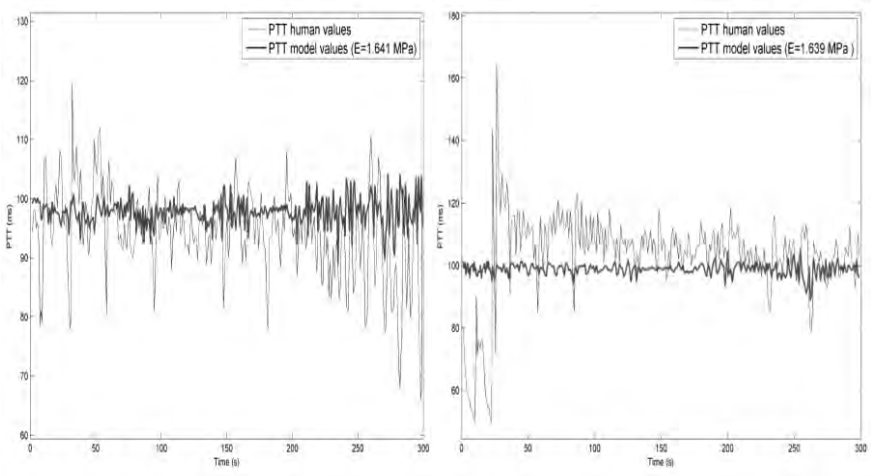
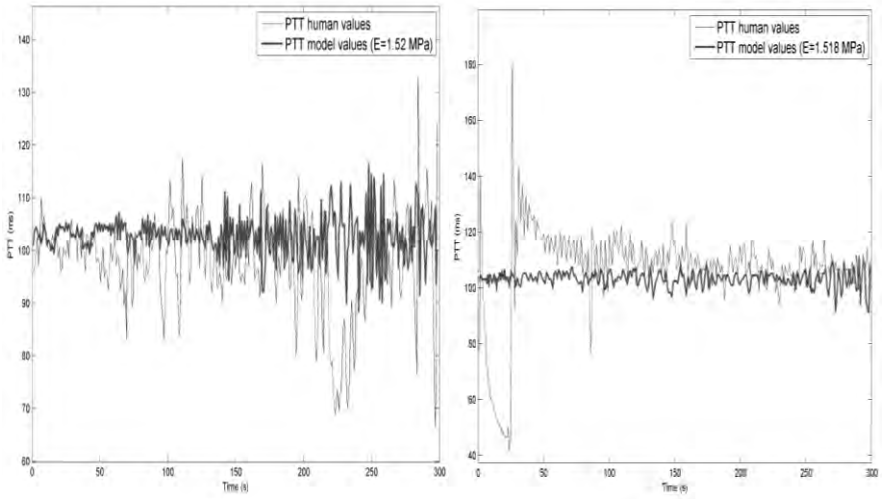


Fig. 13. The fitting results of modules E5, E_6 for Subject_2: a) Fitting result of first pattern breathing; b) Fitting result of second pattern breathing



a) b)

Fig. 14. The fitting results of modules E_7, E_8 for Subject_3: a) Fitting result of first pattern breathing; b) Fitting result of second pattern breathing

It can be seen from figures 12-14, that for each particular subject with a few pattern breathing it is possible to determine approximately one Young's elasticity module E which is very important for estimation of arterial wall condition. Subject_1 has approximately elasticity module E=1.02 MPa, Subject_2 has E=1.64 MPa and Subject_3 has E=1.52 MPa which are in physiological range of the elasticity modules. Further research is necessary to connect more subject and patient data with PTT measurements, like sex, age, history of disease, blood analysis etc.

5. Discussion and conclusions

The proposed model is simplification of simulation for the complex pulsatile flow through arterial system and non-invasive measurement of PTT. The initial results have shown good correlation and validation for estimation of the arterial stiffness. The assumption of the elastic behavior of the walls is used which is simplification of the nonlinear compliance of the arterial wall. Even with this approximation good results are achieved. Stress distribution inside the wall can be varied with different combination of peripheral resistance, compliance. Also shear stress distribution which is crucial factor for activation of endothelial cells and atherosclerosis disease can be fitted with different change of blood viscosity and concentration of LDL and HDL in the standard blood patient analysis. Using of the administering drugs that alter nitric oxide synthesis in the endothelial cells lining blood vessels can be also easy implemented in this model. Primary aim of this study was to evaluate

feasibility of estimation of arterial stiffness using PTT measurements. We measure PTT for different breathing rhythms of the three subjects and compared measurements with simplified finite element fluid-structure interaction model. Numerical simulation analyzed a fluid-structure interaction where fluid and solid domain represent the straight deformable artery segment. The length of numerical model approximately corresponds to the distance from the main aorta root to the finger measurement position of the subject. The simulation results using this simplified approach provide a very good fit with the elastic property of the arterial wall. Therefore, our preliminary results indicate that pulse travel time measurement can be used for noninvasive assessment of the arterial stiffness. We are aware that isotropic assumption of the arterial wall is just first approximation and our future research will go in direction to include more complex component materials. Difference in time-history of measured and simulated PPT values is also due to isotropic assumption of the arterial wall. This is an extension of our work in [18] in terms of more subjects in the experiment and flow-chart for CFD program as well as the algorithm for coupled problem fluid-structure interaction are described in details.

The proposed method could allow the implementation of screening diagnostics. For clinical using it is sufficient to register equal duration of ECG signal and the distal arterial pulse, which are carried out with non-invasive methods by means of widely available monitoring devices.

Acknowledgnments. This work has been partly supported by Ministry of Education and Science of Serbia, Grant No. III-41007, OI-174028 and FP7 ICT-2007-2-5.3 (224297) ARTreat project.

References

1. Allen, J.: Photoplethysmography ant its application in clinical physiological measurement. Physiological measurement, Vol. 28, No. 3. (2007)
2. Cox, P.O., Madsen, C., Ryan, K. L., Convertino, V. A., Jovanov, E.: Investigation of Photoplethysmogram Morphology for the Detection of Hypovolemic States, 30th Annual International IEE EMBS conference, Vancuver, BC. 5486 – 5489. (2008)
3. McDonald, Nichols, W.W., O'Rouke, M.F.: McDonald's blood flow in arteries: theoretical, experimental and clinical principles. Oxford University Press. (1988)
4. Steptoe, A.: Pulse wave velocity and blood pressure changes: calibration and applications. Psychophysiology, Vol. 13, 488-493. (1976)
5. Callaghan, FJ., Geddes, LA., Babbs, CF., Bourland, JD.: Relationship between pulse-wave velocity and arterial elasticity. Med Biol Eng Comp, Vol. 24, No. 3. 248–254. (1986)
6. Laurent, S., Boutouyrie, P., Asmar, R.: Aortic stiffness is an independent predictor of all-cause and cardiovascular mortality in hypertensive patients. Hypertension, Vol. 37. 1236–1241. (2001)
7. Constant, I., Laude, D., Murat, I.: Pulse rate variability is not a surrogate for heart rate variability. Clinical Science, Vol. 97. 391–397. (1999)

8. Drinnan, M.J.: Relation between heart rate and pulse transit time during paced respiration. Physiology Measurement, Vol. 22. 425–432. (2001)
9. Giardino, N.D.: Comparison of finger plethysmograph to ECG in the measurement of heart rate variability. Psychophysiology, Vol. 39. 246-252. (2002)
10. Johansson, A.: Estimation of respiratory volumes from the photoplethysmographic signal part.1: experimental results. Med.Biol.Eng. Comput., Vol. 37. 42-47. (1999)
11. http://www.nonin.com/
12. Wong, Y.M., Zhang, Y.T., The effects of exercises on the relationship between pulse transit time and arterial blood pressure. Conf. of IEEE Engineering in Medicine and Biology Society, Vol. 5 5576-5578. (2005)
13. Filipovic, N., Ivanovic, M., Krstajic, D., Kojic, M.: Hemodynamic flow modeling through an abdominal aorta aneurysm using data mining tools. IEEE Transactions on Information Technology in BioMedicine, Vol. 15, No. 2, 189-194. (2011)
14. Filipovic N, Milasinovic D, Zdravkovic N, Böckler D, von Tengg-Kobligk H.: Impact of aortic repair based on flow field computer simulation within the thoracic aorta. Computer Methods and Programs in Biomedicine, Vol. 101, No. 3, 243-52. (2011)
15. Kojic M, Filipovic N, Zivkovic M, Slavkovic R, Grujovic N: PAK Finite Element Program. Faculty of Mech. Engrg, University of Kragujevac, Serbia. (2010)
16. Filipovic N., Mijailovic S., Tsuda A. and Kojic M.: An Implicit Algorithm Within The Arbitrary Lagrangian-Eulerian Formulation for Solving Incompressible Fluid Flow With Large Boundary Motions. Comp. Meth. Appl. Mech. Eng. Vol 195. 6347-6361. (2006)
17. Nelder, J., Mead, R.: A simplex method for function minimization. Computer Journal Vol. 7, No.2, 308-313. (1965)
18. Peulic A., Jovanov, E., Radovic, M., Saveljic, I., Zdravkovic, N. Filipovic, N.: Arterial Stiffness modeling using variations of Pulse Transit Time. 10th BioEng, 5-7 October, Kos, Greece (2011).

Aleksandar Peulić received the Diploma degree in electronic engineering from Faculty of Electronic Engineering, University of Nis, Nis, Serbia, in 1994, the Master of Science in electrical engineering from Faculty of Electronic Engineering, University of Nis, Nis, Serbia, in 2000 and the Ph.D. degree in electrical engineering from the University of Kragujevac, Kragujevac, Serbia, in 2007. From 2005 to 2007, he was part time with the University of Maribor as doctoral research. From 2008 to 2009, he was as postdoctoral at the University of Alabama in Huntsville. His research interests include microcontrollers systems, wearable sensors, bioengineering, software engineering, and computer control systems.

Nataša Milojević graduated at Faculty of Mechanical Engineering, University of Belgrade, Serbia, in 1995. Received the Diploma degree for Master of Science at the same University, in the area of Bioengineering. Currently works on the Ph.D. From 1995 to 1997, worked as assistant at Faculty of Mechanical Engineering, University of Belgrade. From 1997 until

now works in the Intellectual Property Office of Serbia as Counselor for patents. Interested in bioengineering, patent searching and process engineering.

Emil Jovanov is an Associate Professor in the Electrical and Computer Engineering Department at the University of Alabama in Huntsville. He received his Dipl. Ing., MSc, and PhD from the University of Belgrade. He is recognized as the originator of the concept of wireless body area networks for health monitoring and he is one of the leaders in the field of wearable health monitoring. His research interests include wearable health monitoring, ubiquitous and mobile computing, and biomedical signal processing

Miloš Radović is PhD student at Faculty of Engineering in University of Kragujevac. His research interests include data mining, medical imaging reconstruction and nonlinear parameter estimation.

Igor Saveljić is PhD student at Faculty of Engineering in University of Kragujevac. His research interests include 3D mesh generation, finite element method, fluid-structure interaction and nonlinear parameter estimation.

Nebojša Zdravković received the Ph.D. in bioengineering from the University of Kragujevac, Serbia in 2000. He is currently a Associate Professor in Informatics and Medical Statistics at Faculty of Medical Science, University of Kragujevac, Serbia His research interests are in data mining, statistical methods, finite element method fluid mechanics, coupled problems

Nenad Filipović received the Ph.D. in bioengineering from the University of Kragujevac, Serbia in 1999. He was Research Associate at Harvard School of Public Health in Boston, USA. He is currently a Professor in Bioengineering at Faculty of Engineering, University of Kragujevac, Serbia. His research interests are in the area of fluid mechanics, coupled problems; fluid-structure interaction, heat transfer; biofluid mechanics; biomechanics, multi-scale modeling, discrete modeling, molecular dynamics, computational chemistry and bioprocess modeling. He is author and co-author 6 textbooks and 1 monograph on English language, over 50 publications in peer review journals and over 5 software for modeling with finite element method and discrete methods from fluid mechanics and multiphysics. He leads a number of national and international projects in area of bioengineering.

Received: May 31, 2012; Accepted: December 18, 2012.

CIP – Каталогизација у публикацији
Народна библиотека Србије, Београд

004

COMPUTER Science and Information Systems : the International journal / Editor-in-Chief Mirjana Ivanović. – Vol. 10, No 1 (2013) - . – Novi Sad (Trg D. Obradovića 3): ComSIS Consortium, 2012 - (Belgrade : Sigra star). –30 cm

Polugodišnje. – Tekst na engleskom jeziku

ISSN 1820-0214 = Computer Science and Information Systems
COBISS.SR-ID 112261644

Cover design: V. Štavljanin
Printed by: Sigra star, Belgrade

ComSIS Computer Science and Information Systems
Volume 10, Number 1, January 2013

Contents

ISSN: 1820-0214