

Wojciech Zamojski
Jacek Mazurkiewicz
Jarosław Sugier
Tomasz Walkowiak
Janusz Kacprzyk (Eds.)

New Results in Dependability and Computer Systems

Proceedings of the 8th International Conference
on Dependability and Complex Systems
DepCoS-RELCOMEX, September 9–13, 2013,
Brunów, Poland

Advances in Intelligent Systems and Computing

Volume 224

Series Editor

J. Kacprzyk, Warsaw, Poland

For further volumes:

<http://www.springer.com/series/11156>

Wojciech Zamojski · Jacek Mazurkiewicz
Jarosław Sugier · Tomasz Walkowiak
Janusz Kacprzyk
Editors

New Results in Dependability and Computer Systems

Proceedings of the 8th International
Conference on Dependability and Complex
Systems DepCoS-RELCOMEX,
September 9–13, 2013,
Brunów, Poland

 Springer

Editors

Wojciech Zamojski
Institute of Computer Engineering,
Control and Robotics
Wrocław University of Technology
Wrocław
Poland

Tomasz Walkowiak
Institute of Computer Engineering,
Control and Robotics
Wrocław University of Technology
Wrocław
Poland

Jacek Mazurkiewicz
Institute of Computer Engineering,
Control and Robotics
Wrocław University of Technology
Wrocław Poland

Janusz Kacprzyk
Polish Academy of Sciences,
Systems Research Institute
Warszawa
Poland

Jarosław Sugier
Institute of Computer Engineering,
Control and Robotics
Wrocław University of Technology
Wrocław
Poland

ISSN 2194-5357

ISSN 2194-5365 (electronic)

ISBN 978-3-319-00944-5

ISBN 978-3-319-00945-2 (eBook)

DOI 10.1007/978-3-319-00945-2

Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013940156

© Springer International Publishing Switzerland 2013

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

We are pleased and honoured to present the proceedings of the Eight International Conference on Dependability and Complex Systems DepCoS-RELCOMEX which was held in a beautiful Brunów Palace, Poland, from 9th to 13th September, 2013.

DepCoS – RELCOMEX is an annual conference series organized by the Institute of Computer Engineering, Control and Robotics (CECR), Wrocław University of Technology, since 2006. Its idea came from the heritage of the other two cycles of events: RELCOMEX (1977–89) and Microcomputer Schools (1985–95) which were organized by the Institute of Engineering Cybernetics (the previous name of CECR) under the leadership of prof. Wojciech Zamojski, now also the DepCoS chairman. In contrast to those previous events focused on the classical reliability analysis, the DepCoS mission is to promote a more comprehensive approach which in the new century has earned the name *dependability*.

Contemporary technical systems are integrated unities of technical, information, organization, software and human (users, administrators and management) resources. Their complexity stems not only from involved technical and organization structures (comprising both hardware and software resources) but also from complexity of information processes (processing, monitoring, management, etc.) realized in their operational environment. With system resources being dynamically allocated to the on-going tasks, a flow of system events (incoming and/or on-going tasks, decisions of a management system, system faults, defensive system reactions, etc.) may be modelled as a deterministic or/and probabilistic event stream. Complexity and multiplicity of processes, their concurrency and their reliance on the in-system intelligence (human and artificial) significantly impedes the construction of strict mathematical models and limits evaluation of adequate system measures. In many cases, analysis of modern complex systems is confined to quantitative studies (e.g. Monte Carlo simulations) which prevents development of appropriate methods of system design and selection of policies for system exploitation. Security and confidentiality of information processing

introduce further complications into the system models and the evaluation methods.

Dependability tries to deal with all those challenges by employing a multidisciplinary approach to theory, technology and maintenance of systems working in a real (and very often unfriendly) environment. As opposed to “classic” reliability which focuses mainly on technical system resources (components and structures built from them), dependability studies investigate the system as a multifaceted and sophisticated amalgamation of technical, information and also human resources concentrating on efficient realization of tasks, services and jobs in such an environment. Traditional methods of reliability evaluation focused mainly on technical levels are usually insufficient and more innovative methods of dependability analysis, often based on intelligent and soft computing algorithms, need to be applied. The 50 submissions selected for this volume illustrate the wide diversity of problems that need to be explored, for example methodologies and practical tools for modelling, design and simulation of the systems, security and confidentiality in information processing, specific issues of heterogeneous, today often wireless, computer networks, management of transportation networks, etc.

In the closing words of this introduction we would like to emphasize the role of all reviewers whose support helped to refine the contents of this volume. The following people took active part in the evaluation process of the conference submissions: Salem Abdel-Badeeh, Andrzej Białas, Frank Coolen, Manuel Gil Perez, Zbigniew Huzar, Vyacheslav Kharchenko, Alexey Lastovetsky, Marek Litwin, Jan Magott, István Majzik, Jacek Mazurkiewicz, Yiannis Papadopoulos, Oksana Pomorova, Krzysztof Sacha, Mirosław Siergiejczyk, Ruslan Smeliansky, Janusz Sosnowski, Jarosław Sugier, Victor Toporkov, Tomasz Walkowiak, Max Walter, Bernd E. Wolfinger, Marina Yashina, Wojciech Zamojski, and Włodzimierz Zuberek.

Finally, we would like to express our sincere gratitude to the authors of all the works selected for publication – we hope that their submissions will be interesting to scientists, researchers, practitioners and students who investigate dependability problems in computer systems and networks, and in their diverse applications.

The Editors

Eight International Conference on Dependability and Complex Systems DepCoS-RELCOMEX

organized by

Institute of Computer Engineering, Control and Robotics,
Wrocław University of Technology
under the auspices of prof. Tadeusz Więckowski, Rector

Brunów Palace, Poland, 9–13 September 2013

Program Committee

Wojciech Zamojski (Chairman)
*Wrocław University of Technology,
Poland*

Ali Al-Dahoud
*Al-Zaytoonah University,
Amman, Jordan*

Salem Abdel-Badeeh
*Ain Shams University Abbasia,
Cairo, Egypt*

George Anders
University of Toronto, Canada

Włodzimierz M. Barański
*Wrocław University of Technology,
Poland*

Andrzej Białaś
*Institute of Innovative Technologies
EMAG, Katowice, Poland*

Dariusz Caban
*Wrocław University of Technology,
Poland*

Krzysztof Cios
*Virginia Commonwealth University,
Richmond, USA*

Frank Coolen
Durham University, UK

Antonio Ferrari
University of Aveiro, Portugal

Francesco Flammini
*University of Naples “Federico II”,
Italy*

Manuel Gill Perez
University of Murcia, Spain

Janusz Górski
*Gdańsk University of Technology,
Poland*

Franciszek Grabski
Naval University, Gdynia, Poland

Zbigniew Huzar
*Wrocław University of Technology,
Poland*

Igor Kabashkin
*Transport and Telecommunication
Institute, Riga, Latvia*

Janusz Kacprzyk
*Polish Academy of Sciences,
Warsaw, Poland*

Andrzej Kasprzak

*Wrocław University of Technology,
Poland*

Vyacheslav S. Kharchenko

*National Aerospace University
“KhAI”, Kharkov, Ukraine*

Mieczysław M. Kokar

*Northeastern University, Boston,
USA*

Krzysztof Kołowrocki

*Gdynia Maritime University,
Poland*

Leszek Kotulski

*AGH University of Science and
Technology, Kraków, Poland*

Henryk Krawczyk

*Gdańsk University of Technology,
Poland*

Alexey Lastovetsky

University College Dublin, Ireland

Elena Liapuntsova

*Moscow State University of Railway
Engineering, Russia*

Marek Litwin

ITS Poland

Jan Magott

*Wrocław University of Technology,
Poland*

Istvan Majzik

*Budapest University of Technology
and Economics, Hungary*

Jacek Mazurkiewicz

*Wrocław University of Technology,
Poland*

Katarzyna M. Nowak

*Wrocław University of Technology,
Poland*

Sergey Orlov

*Transport and Telecommunication
Institute, Riga, Latvia*

Yiannis Papadopoulos

Hull University, UK

Oksana Pomorova

*Khmelnytsky National University,
Ukraine*

Ewaryst Rafajłowicz

*Wrocław University of Technology,
Poland*

Krzysztof Sacha

*Warsaw University of Technology,
Poland*

Mirosław Siergiejczyk

*Warsaw University of Technology,
Poland*

Yurij Skobtsov

*Donetsk National Technical
University, Donetsk, Ukraine*

Ruslan Smeliansky

Moscow State University, Russia

Czesław Smutnicki

*Wrocław University of Technology,
Poland*

Janusz Sosnowski

*Warsaw University of Technology,
Poland*

Jarosław Sugier

*Wrocław University of Technology,
Poland*

Ryszard Tadeusiewicz

*AGH University of Science and
Technology, Kraków, Poland*

Victor Toporkov

*Moscow Power Engineering Institute
(Technical University), Russia*

Tomasz Walkowiak

*Wrocław University of Technology,
Poland*

Max Walter

Siemens, Germany

Bernd E. Wolfinger

University of Hamburg, Germany

Stanisław Wrycza

University of Gdańsk, Poland

Marina Yashina

*Moscow Technical University of
Communication and Informatics,
Russia*

Irina Yatskiv

*Transport and Telecommunication
Institute, Riga, Latvia*

Jan Zarzycki

*Wrocław University of Technology,
Poland*

Włodzimierz Zuberek

*Memorial University, St. John's,
Canada*

Organizing Committee

Wojciech Zamojski (Chairman)

Włodzimierz M. Barański

Jacek Mazurkiewicz

Katarzyna M. Nowak

Jarosław Sugier

Tomasz Walkowiak

Contents

Application Level Execution Model for Transparent Distributed Computing	1
<i>Razvan-Mihai Aciu, Horia Ciocarlie</i>	
Software Support of the Risk Reduction Assessment in the ValueSec Project Flood Use Case	11
<i>Jacek Bagiński</i>	
Risk Assessment Aspects in Mastering the Value Function of Security Measures	25
<i>Andrzej Bialas</i>	
Reduction of Computational Cost in Mutation Testing by Sampling Mutants	41
<i>Iлона Bluemke, Karol Kulesza</i>	
Use of Neural Network Algorithms in Prediction of XLPE HV Insulation Properties under Thermal Aging	53
<i>Boukezzi Larbi, Boubakeur Ahmed</i>	
Computer Simulation Analysis of Cluster Model of Totally-Connected Flows on the Chain Mail	63
<i>Alexander P. Buslaev, Pavel M. Strusinskiy</i>	
Assessment of Network Coding Mechanism for the Network Protocol Stack 802.15.4/6LoWPAN	75
<i>Michał Bylak, Dariusz Laskowski</i>	
Reliability Analysis of Discrete Transportation Systems Using Critical States	83
<i>Dariusz Caban, Tomasz Walkowiak</i>	

A Reference Model for the Selection of Open Source Tools for Requirements Management	93
<i>Bartosz Chrabski, Cezary Orłowski</i>	
A Probabilistic Approach to the Count-To-Infinity Problem in Distance-Vector Routing Algorithms	109
<i>Adam Czubak</i>	
A Quality Estimation of Mutation Clustering in C# Programs	119
<i>Anna Derezińska</i>	
Using Virtualization Technology for Fault-Tolerant Replication in LAN	131
<i>Fernando Dettoni, Lau Cheuk Lung, Aldelir Fernando Luiz</i>	
Quantification of Simultaneous-AND Gates in Temporal Fault Trees	141
<i>Ernest Edifor, Martin Walker, Neil Gordon</i>	
Improving of Non-Interactive Zero-Knowledge Arguments Using Oblivious Transfer	153
<i>Alexander Frolov</i>	
Virtual Environment for Implementation and Testing Private Wide Area Network Solutions	173
<i>Mariusz Gola, Adam Czubak</i>	
Optimization of Privacy Preserving Mechanisms in Mining Continuous Patterns	183
<i>Marcin Gorawski, Pawel Jureczek</i>	
Technical and Program Aspects on Monitoring of Highway Flows (Case Study of Moscow City)	195
<i>M.G. Gorodnichev, A.N. Nigmatulin</i>	
Integral Functionals of semi-Markov Processes in Reliability Problems	205
<i>Franciszek Grabski</i>	
Generating Repair Rules for Database Integrity Maintenance	215
<i>Feras Hanandeh, Yaser Quasmeh</i>	
Optimization Algorithm for the Preservation of Sensor Coverage	225
<i>Codruta-Mihaela Istin, Horia Ciocarlie, Razvan Aciu</i>	

Methods for Detecting and Analyzing Hidden FAT32 Volumes Created with the Use of Cryptographic Tools 237
Ireneusz Józwiak, Michał Kędziora, Aleksandra Melnińska

Critical Infrastructures Safety Assessment Combining Fuzzy Models and Bayesian Belief Network under Uncertainties 245
Vyacheslav Kharchenko, Eugene Brezhniev, Vladimir Sklyar, Artem Boyarchuk

Towards Evolution Methodology for Service-Oriented Systems 255
Szymon Kijas, Andrzej Zalewski

The LVA-Index in Clustering 275
Piotr Lasek

Three Different Approaches in Pedestrian Dynamics Modeling – A Case Study 285
Robert Lubaś, Janusz Miller, Marcin Mycek, Jakub Porzycki, Jarosław Wąs

The End-To-End Rate Adaptation Application for Real-Time Video Monitoring 295
P. Lubkowski, Dariusz Laskowski

Discrete Transportation Systems Quality Performance Analysis by Critical States Detection 307
Jacek Mazurkiewicz, Tomasz Walkowiak

An Expanded Concept of the Borrowed Time as a Mean of Increasing the Average Speed Isotropy on Regular Grids 315
Marcin Mycek

Freshness Constraints in the RT Framework 325
Wojciech Pikulski, Krzysztof Sacha

Transformational Modeling of BPMN Business Process in SOA Context 335
Andrzej Ratkowski

Algorithmic and Information Aspects of the Generalized Transportation Problem for Linear Objects on a Two Dimensional Lattice 345
Anton Shmakov

Reliability Assessment of Supporting Satellite System EGNOS	353
<i>Miroslaw Siergiejczyk, Adam Rosiński, Karolina Krzykowska</i>	
Vbam – Byzantine Atomic Multicast in LAN Based on Virtualization Technology	365
<i>Marcelo Ribeiro Xavier Silva, Lau Cheuk Lung, Leandro Quibem Magnabosco, Luciana de Oliveira Rech</i>	
An Approach to Automated Verification of Multi-Level Security System Models	375
<i>Andrzej Stasiak, Zbigniew Zieliński</i>	
A Web Service-Based Platform for Distributed Web Applications Integration	389
<i>Pawel Stelmach, Lukasz Falas</i>	
Universal Platform for Composite Data Stream Processing Services Management	399
<i>Pawel Stelmach, Patryk Schauer, Adam Kokot, Maciej Demkiewicz</i>	
Proposal of Cost-Effective Tenant-Based Resource Allocation Model for a SaaS System	409
<i>Wojciech Stolarz, Marek Woda</i>	
Automatic Load Testing of Web Application in SaaS Model	421
<i>Emil Stupiec, Tomasz Walkowiak</i>	
Implementing Salsa20 vs. AES and Serpent Ciphers in Popular-Grade FPGA Devices	431
<i>Jaroslaw Sugier</i>	
On Testing Wireless Sensor Networks	439
<i>Tomasz Surmacz, Bartosz Wojciechowski, Maciej Nikodem, Mariusz Stabicki</i>	
Towards Precise Architectural Decision Models	449
<i>Marcin Szlenk</i>	
Slot Selection Algorithms for Economic Scheduling in Distributed Computing with High QoS Rates	459
<i>Victor Toporkov, Anna Toporkova, Alexey Tselishchev, Dmitry Yemelyanov</i>	
K-Induction Based Verification of Real-Time Safety Critical Systems	469
<i>Tamás Tóth, András Vörös, István Majzik</i>	

Native Support for Modbus RTU Protocol in Snort Intrusion Detection System	479
<i>Wojciech Tylman</i>	
SCADA Intrusion Detection Based on Modelling of Allowed Communication Patterns	489
<i>Wojciech Tylman</i>	
System for Estimation of Patient’s State – Discussion of the Approach	501
<i>Wojciech Tylman, Tomasz Waszyrowski, Andrzej Napieralski, Marek Kamiński, Zbigniew Kulesza, Rafał Kotas, Paweł Marciniak, Radosław Tomala, Maciej Wenerski</i>	
Dependability Aspects of Autonomic Cooperative Computing Systems	513
<i>Michał Wódczak</i>	
Life Cycle Cost through Reliability	523
<i>Manac’h Yann-Guirec, Benfriha Khaled, Aoussat Améziane</i>	
Verification of Infocommunication System Components for Modeling and Control of Saturated Traffic in Megalopolis ...	531
<i>Marina V. Yashina, Andrew V. Provorov</i>	
Shuffle-Based Verification of Component Compatibility	543
<i>W.M. Zuberek</i>	
Author Index	553

Application Level Execution Model for Transparent Distributed Computing

Razvan-Mihai Aciu and Horia Ciocarlie

Department of Computer and Software Engineering, "Politehnica" University of Timisoara
Blvd Vasile Parvan, Nr. 2, Postcode 300223, Timisoara, Romania
razvanaciu@yahoo.com, horia@cs.upt.ro

Abstract. Writing a distributed application involves using a number of different protocols and libraries such as CORBA, MPI, OpenMP or portable virtual machines like JVM or .NET. These are independent pieces of software and gluing them together adds complexity which can be error prone. Still, some issues such as transparent creation and synchronization of the parallel distributed threads, code replication, data communication and hardware and software platform abstraction are not yet fully addressed. For these reasons a programmer must still manually handle tasks that should be automatically and transparently done by the system. In this work we propose a novel computing model especially designed to abstract and automate the distributed computing requirements ensuring at the same time the dependability and scalability of the system. Our model is designed for a portable virtual machine suitable to be implemented both on hardware native instruction set as well as in other virtual machines like JVM or .NET to ensure its portability across hardware and software frameworks.

1 Introduction

Distributed computing is a domain with intense research and applicability in many areas like biology [1-2], physics [3-4], agriculture [5], computing [6]. As the computer networks are increasingly popular, there are more and more possibilities to access data storage and computation power which allows solving of problem types earlier accessible only to supercomputers or dedicated data centers. The main parallelization opportunities today are represented by CPUs with multiple cores, distributed computing in a network and specialized computing using GPUs.

We focus mainly on showing a reliable model for transparent distributed computing, capable to handle by itself almost all the low level details needed by network code replication, data communication and distributed thread synchronization. We demonstrate that our model insures hardware and software abstraction, it scales well with the available computing resources and it is sufficiently general to handle other aspects involved in heterogeneous distributed computing such as different operating systems or abstracting the execution on CPU cores or in a network.

For applications that run on a single machine, the task of writing a parallel algorithm is simplified by the fact that many aspects involved are the same, for example the memory layout and access, threads creation and synchronization, computing units binary code. There are some models and libraries like OpenMP that address this situation using special preprocessor instructions or annotations to instruct the compiler to automatically parallelize a loop [7-8]. At the same time many modern additions to standard libraries provide high level concepts for parallel computing, for example thread pools.

When the same algorithm is implemented for distributed computing new problems arise. We mention different data layout and instruction, different operating systems and possible network failures. To address the above issues different standards and libraries were proposed: CORBA, MPI, Java RMI, DCOM and Ibis [8]. Most of these are low level protocols that try to hide the distributed platform differences, but they are not sufficiently high level to hide from the programmer details like code replication or data synchronization [8-9].

Another fundamental issue for any distributed system is its scalability, due to the fact that the system should make an optimal use of all its computing resources [10]. Heterogeneous resources are available in many different versions, speeds or instruction sets. This makes hard for the programmer to make an optimal use of these resources, in order that their combined workload to lead to a minimal application processing time or to another desired target [11]. At the same time some resources can have a dynamic behavior, because they can be added or removed from the system by request or due to network errors.

When analyzing these aspects, it can be seen that most of the work involved in writing distributed software is in fact necessary to address repetitive and standard tasks. All these tasks can be automatically addressed by using a proper infrastructure and model. Such an application level model greatly improves the software dependability. We present such a model, motivate our design decisions and show the results of one of its possible practical implementations.

2 Application Level Distributed Computing Model

Ideally speaking, a transparent application level model should hide from the programmer any low level task. At the same time, it should fit with only minimal additions to the existing programming languages and frameworks, so it can be easily implemented. Our model involves only three concepts, close to OOP programming style and it should seem familiar to any programmer with an OOP background. We will present it by using an example.

We chose to implement the well known Mandelbrot set on a given interval. The example is written in a C++-like language and shows the relevant code for our model:


```

unit MandelbrotLine{
  double xMin,xMax;
  int width,maxIterations;
  MandelbrotLine(double xMin,double xMax,int width,int maxIterations)
  {
    this->xMin=xMin;
    this->xMax=xMax;
    this->width=width;
    this->maxIterations=maxIterations;
  }
  string run(double y)
  {
    int pixel,iteration;
    stringstream r;
    double xi,yi,xb,xtemp;
    for(pixel=0;pixel<width;pixel++){
      xb=xMin+pixel*(xMax-xMin)/width;
      xi=yi=0;
      iteration=0;
      while(xi*xi+yi*yi<2*2&& iteration<maxIterations){
        xtemp=xi*xi-yi*yi+xb;
        yi=2*xi*yi+y;
        xi=xtemp;
        iteration++;
      }
      r<<iteration%256<<" "; //to gray tones
    }
    return r.str();
  }
}

#define WIDTH 1000
#define HEIGHT 1000
#define MAX_ITER 10000
int main()
{
  int lineIdx;
  double yLine;
  string img[HEIGHT];
  double xMin=0.33072017,xMax=0.33925741;
  double yMin=0.04369091,yMax=0.0522281593;
  with(img;MandelbrotLine(xMin,xMax,WIDTH,MAX_ITER)){
    for(lineIdx=0;lineIdx<HEIGHT;lineIdx++){
      yLine=yMin+lineIdx*(yMax-yMin)/HEIGHT;
      run[lineIdx](yLine);
    }
  }
  ofstream file("mandelbrot.pgm");
  file<<"P2"<<endl<<WIDTH<<" "<<HEIGHT<<endl<<"255"<<endl;
  for(lineIdx=0;lineIdx<HEIGHT;lineIdx=lineIdx+1)
    file<<img[lineIdx]<<endl;
}

```

Fig. 1 Model example for a distributed algorithm

For each image line a new invocation is created. An invocation encapsulates all data needed for a single computation and it is asynchronously run on a separate distributed thread when a computing resource becomes available.

In our model, a thread is always associated with a resource (the existent cores in network) and the maximum number of threads is at most equal to the number of resources. In this way we eliminate the unnecessary task switching and at the same time we use all the resource at their maximum capacity. If the number of invocations exceeds the available resources, the invocations are put in a queue, waiting for resources to become available. We use the term “invocation” for a scheduled computation and the term “thread” for running invocation.

The example program computes distributedly all the invocations, waits for all of them to complete and writes the results into a file. We used three special concepts to make this program distributed. These concepts are detailed below, each one with its own semantics and requirements.

1) *The “unit” concept*: is the main encapsulation block for an invocation. Like a regular class from the OOP languages it can contain attributes and methods. When used, a “unit” can be run locally, on the same machine, or it can be transparently sent to another computer from the network. There are some significant differences between a “unit” and a regular class. These differences and their rationale are as follows:

The “unit” constructors are used to set the initial state for all the created threads. The constructors are called in the “with” statement (explained in II.B) and the same data is used to initialize all the threads. In our example all the threads have the same x -axis interval $[xMin, xMax]$, the horizontal resolution *width* and the maximum number of iterations, *maxIterations*.

The “unit” “run” methods are used to specify the code for threads. Their parameters are initialized from the scheduled invocations and their result is returned to the caller after the thread ends. In our example every thread needs only its y position on the y -axis to compute a particular line.

The main difference between the constructor data and the invocation data comes from the fact that the constructor data is the same for all invocations, so it can be sent only once for every machine, no matter how many invocations will run on that specific machine. Instead, an invocation data needs to be specifically sent for each thread.

A “unit” can use extern functions or other types such as classes, but it cannot access, directly or indirectly, extern data. If it accesses functions or classes, these are automatically packed with it and sent to the remote machine. The rationale for not allowing extern data access is that it would require a lot of slow and unreliable network traffic, including data serialization and synchronization with other threads. If the programmer will try to use from a remote “unit” some external data in the same way as a regular local data, this will greatly slowdown the entire thread and will also create a bottleneck for all the other (possibly remote) threads waiting to access that data. Because of this, it was chosen for every thread to be able to access only its own data and this data is sent along with it.

2) *The “with” concept*: is an encapsulation block for creating and synchronizing invocations. “with” accepts two or three parameters, separated by semicolons. The first one is a destination for the threads results, the second is a “unit” constructor and the third one, which is optional, is a set of flags to control different aspects of the statement.

The destination can be an n-dimensional array, an object with a special interface or a function/closure. In case of an array, it will hold after the execution of the “with” block at the positions given by the invocations ordinals (explained at the “for” concept) the results of the invocations. If objects are used as destination, they must implement a specific interface with a handler method. In case of functions/closures, they will receive the invocations results and ordinals as parameters.

The “unit” constructor is used to specify what “unit” will be used (only one for a “with” statement) and its initial data. This data will be the same for all threads creation. The optional “flags” parameter is a set of flags used to specify different aspects, for example the restriction to use only local cores (no network traffic), to enable/disable the GPU processing, etc.

“with” ensures on its end the computation of all invocations and the synchronization of all the running threads, waiting for them to complete, similar with the “join” functions used in multithreading programming. The underlying framework is also responsible with the “unit” code replication in network, data serialization for invocations parameters and results and automatic rerun for computations lost due to network errors. The “with” statement also acts as a threads pool and it monitors the status of the processing units (CPU, GPU, network computers), assigning new invocations to them when there are free resources.

3) *The “run” concept:* can be used only inside “with” and consists of two lexical parts. The first part is represented like an n-dimensional array access and it is used to specify the ordinals of the invocation. The second part is represented like a function call and it is used to specify invocation specific parameters to be passed to the unit “run” methods.

The invocation ordinals are used to specify unique ids for every invocation. Our Mandelbrot example uses only one ordinal, which is the index of the line returned by the invocation. The parameters specified in the second part of the “run” statement will be used on that specific invocation run.

The “run” statement creates and stores an invocation for the threads pool provided by the encapsulating “with”. In this way the system can optimally choose how to run the invocations, one by one or in batches (for GPUs).

3 Model Performance

To assess the theoretical performance of our model, we consider N_C computers in the distributed system with a total of N_P processing units (cores) which need to execute a number of N_I invocations ($N_P \leq N_I$), each invocation requiring a T_I time to complete and T_S is the time to setup one computer (send the “unit” to it):

$$T_T = T_S + \lceil N_I / N_P \rceil * T_I, \quad (1)$$

where T_T is the total computation time and $\lceil x \rceil = \min\{n \in \mathbb{Z} \mid n \geq x\}$ (2)

$$\text{if } N_P \rightarrow N_I: \quad \lim_{N_P \rightarrow N_I} T_T = T_S + T_I \quad (3)$$

In that case the total execution time for all distributed threads is the execution time of the longest thread (if there are heterogeneous resources) and the setup time for

the remote computers, which depends on factors like the size of the “unit” data and network usage.

$$\text{As } T_I \text{ can also be seen as } T_I = T_{IN} + T_{IC} \quad (4)$$

where T_{IN} is the time needed for network operation (sending the “run” parameters to the processing unit and receiving the results from it) and T_{IC} is the effective computation time on the processing unit, in the best case

$$(N_P = N_I): T_T = T_S + T_{IN} + T_{IC} \quad (5)$$

Because $T_S + T_{IN}$ depends only on the network performance, for optimal results it is best that it’s weight with respect to the overall computing time to be as small as possible. The optimal case is when the T_{IC} of the distributed threads is much larger than $T_S + T_{IN}$, so the distributed system spends most of its time in doing the effective computation than on network traffic. In this case, the performance of the distributed computation becomes close to the performance of executing all the invocations locally on a machine with N_I cores.

4 Implementation and Study

We implemented our model by developing a special virtual machine (VM) and associated runtime, capable to run the VM on CPU cores. For every machine all the computing resources are abstracted using a server which can receive a “unit” and invocations to be run on it. An application runs as a client and it makes requests to the available servers when it needs to run multiple threads. The whole process takes place according to the following steps:

1) *The client checks for the available servers:* A list of network hosts is used and every machine is queried about its server version, protocol and the number of available cores.

2) *The client runs the application:* We designed a register based, strongly typed VM with high level abstractions like functions and classes and automatic memory management. Having a portable VM which acts like an abstraction layer between the application and the host available capabilities (native execution using the CPU instruction set or execution on a particular VM like JVM or .NET) allows us to use almost any computer, ensuring both hardware and software framework independence.

A register based VM is also very important for running threads on GPU cores. In order to do this in a portable way, the GPU driver (CUDA or OpenCL) must receive a kernel function written in a C/C++ like language. In our case, the functions from inside a “unit” must be recompiled from the VM opcodes to the required language and using a register based VM makes this job easy.

3) *When the application enters a “with” statement, the runtime is invoked to create a scheduler:* The scheduler receives the unit constructor parameters, the receiver and possible options. The constructor parameters are serialized only once, in the beginning, as they remain constant.

A scheduler creates when needed a number of worker threads, each one responsible with the connection to a computing resource on a server. The worker communicates with the server using a socket which is maintained open during all the worker life. On the server side a new connection object is created in a separate thread for each worker, so every connection will run in a CPU core.

Before handling invocations, a worker makes sure that the needed code (the “unit” code and all its dependencies) are available on server. The server is able to cache all the code sent to it, so a “unit” must be sent only once to each server. The arguments for the “unit” constructor are also sent for each worker.

4) *On a “run” statement, a new invocation is added:* The scheduler keeps a list of all the invocations. The invocations are added asynchronously. When an invocation is added, the scheduler checks if there are available workers for that invocation and if not tries to create a new one using the list of the available servers. At most, the total number of workers can be equal with the total number of cores in all servers from the list.

5) *When a worker is free, it processes an invocation from the invocations list:* On server a new VM instance is created on the worker’s connection. The “unit” constructor is called with the initialization data already on server so a new “unit” instance will be available.

The worker sends the specific “run” method signature (to allow method overloading) and its parameters. The “run” method is run on server, isolated on the connection’s specific VM instance.

The server serializes the “run” results and returns them to the worker. If an error occurs during the processing (for example network errors), the worker puts back the invocation in the invocations list for reprocessing. If there is no error, the results are put/sent to the “with” receiver.

In this way, the workers will take invocations from list, run them remotely and put the results where needed. The process continues until there are no invocations left for processing.

6) *At the end of the “with” statement, the scheduler waits for the completion of all invocations:* For this the invocations list must be empty and all workers must have ended their current jobs. After all the invocations are processed, all workers are ended and the scheduler is disposed. On the server side, the resources allocated to the connections are also disposed.

5 Experimental Results

We tested the implementation performance and scalability both on a processor’s local cores and in a network of 10 computers. Our test program is the one from Fig. 1 with HEIGHT=2000, so we have a total of 2000 invocations. The program implementation in our VM resulted in a code set of about 1.1KB and every set were run with a clean server, so on each run the servers invocation setup needed to be complete, by sending each time the required application code to them.

For each test we measured the speedup from the case with 1 core or 1 computer to test the scalability of the distributed computing system and the workload on each core, measured in the percent amount of distributed invocations computed on

that core, to evaluate the implementation capacity to distribute evenly the invocations on all available processing units.

The best scalability would be if the speedup is equal with the number of computers/cores involved in computation from the case of performing the computation on only one computer/core. The best workload distribution would be if all the invocations would be distributed completely equal on all the available processing units, assuming that all the processing units have equal capabilities.

5.1 Tests on a Computer Network

We used a Wi-Fi network of 10+1 computers with 2 cores each and all cores were used. The invocations were allowed to run only on remote computers and one computer was used only to run the main application, so all the created threads can run in equal conditions. We started with one computer and on each test added another computer to the available servers. The results for speedup can be seen in Fig. 2 and for the workload on every core from all computers in Fig. 3.

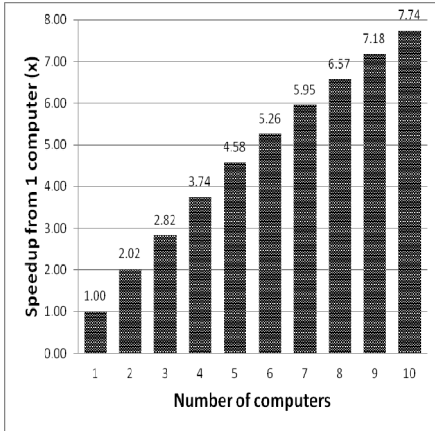


Fig. 2 Speedup results on network

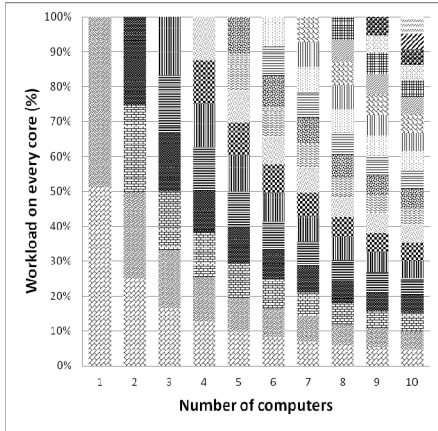


Fig. 3 Workload results on a computer network

When the number of computers is small, so the amount of total computation on each computer is high, the speedup when adding a new one is very close to the optimal case. When the number of computers is higher, the network setup and traffic time, which are constant, starts to have a significant proportion, so the speedup is smaller. This is consistent with our theoretical model performance presented in Section III-D.

Even if we used a Wi-Fi network with lower reliability, the system did a good job in allocating an equal number of invocations on every core. From our test results, the maximal percent difference from the optimum was 13.6%, with an average percent difference of maximum 5% for all test runs.

5.2 Tests on a Computer Cores

We used a computer with 4 cores. We started with one core and on each test we added another core. The results can be seen in Fig. 4 for speedup and in Fig. 5 for the workload on every core.

Because on running invocations only in the local processor cores there is no network traffic involved, the T_S+T_{IN} term from our theoretical model is 0 and only a smaller overhead of threads synchronization is involved. In this case, the speedup has a linear grow, close to the optimal case, both in the beginning and at the ending of the graphic.

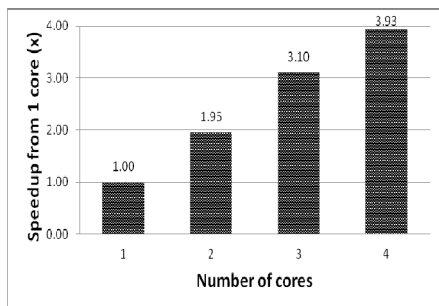


Fig. 4 Speedup results on a computer cores

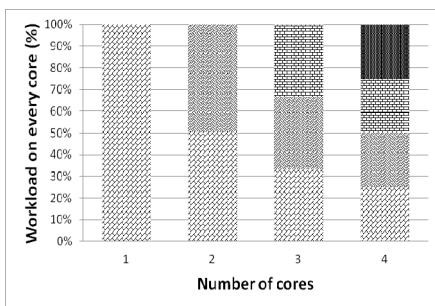


Fig. 5 Workload results on a computer cores

Even if one core was also used to run the main application, the implementation succeeded to distribute the invocations on every core close to the optimal case. From our test results, the maximal percent difference from the optimum was 0.6%, with an average percent difference of maximum 0.6% for all test runs. The small difference from the optimal workload even if one core also runs the “with” scheduler, proves that it mainly waits for the threads to complete, so it takes only very few computing resources.

6 Conclusion

We proposed an application level model for dependable transparent distributed computing. We showed that using only 3 concepts it is possible to model a large domain of distributed algorithms. The model semantics is close to the OOP, which makes the concepts easily to be implemented in most of the actual programming languages.

Our implementation using a portable VM shown that a scheduling system based on distributed thread pools can distribute the invocations computations across all of the available computing resources in a close to optimal manner, obtaining a well balanced workload. It has shown that different computing resources (CPU cores or network computers) can be abstracted using server/client and threads semantics.

The experimental results indicate that the model is scalable, because it succeeded to use well the available resources and the speedups achieved were close to performing the computations as by using multiple programs in parallel.

Even on network failures the application was able to run again the dropped invocations, in order to complete them, so the model is dependable.

We further consider developing our work on directions that include using GPUs as transparent computing resources, ensuring computation reliability and recovery from more possible errors and optimal scheduling algorithms for heterogeneous resources.

References

- [1] Afek, Y., Alon, N., Barad, O., Hornstein, E., Barkai, N., Bar-Joseph, Z.: A Biological Solution to a Fundamental Distributed Computing Problem. *Proc. Natl. Acad. Sci. USA* 108(14), 5488–5491 (2011)
- [2] Macía, J., Posas, F., Solé, R.V.: Distributed computation: the new wave of synthetic biology devices. *Trends in Biotechnology* 30(6), 342–349 (2012)
- [3] Lawrenz, M., Baron, R., Wang, Y., Andrew McCammon, J.: Independent-Trajectory Thermodynamic Integration: A Practical Guide to Protein-Drug Binding Free Energy Calculations Using Distributed Computing. In: *Computational Drug Discovery and Design Methods in Molecular Biology*, vol. 819, pp. 469–486 (2012)
- [4] Charbonneau, A., Agarwal, A., Anderson, M., Armstrong, P., Fransham, K., Gable, I., Harris, D., Impey, R., Leavett-Brown, C., Paterson, M., Podaima, W., Sobie, R.J., Vlie, M.: Data intensive high energy physics analysis in adistributed cloud. In: *Journal of Physics: Conference Series*, vol. 341 (2012)
- [5] Polojärvi, K., Luimula, M., Verronen, P., Pahkasalo, M., Koistinen, M., Tervonen, J.: Distributed System Architectures, Standardization, and Web-Service Solutions in Precision Agriculture. In: *GEOProcessing: The Fourth International Conference on Advanced Geographic Information Systems, Applications, and Services* (2012)
- [6] Jakovits, P., Srirama, S.N., Kromonov, I.: Stratus: A Distributed Computing Framework for Scientific Simulations on the Cloud. In: *IEEE 14th International Conference High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems, HPCC-ICSS* (2012)
- [7] Larsen, P., Ladelsky, R., Karlsson, S., Zaks, A.: Compiler Driven Code Comments and Refactoring. In: *MULTIPROG* (2011)
- [8] Henrio, L., Huet, F., Zsolt, I., Sebestyen, G.: Multi-active Objects. *sop.inria.fr* (2011)
- [9] Lee, H.J., Brown, K.J., Sujeeth, A.K., Chafi, H., Olukotun, K., Rompf, T., Odersky, M.: Implementing Domain-Specific Languages for Heterogeneous Parallel Computing. *IEEE Micro* 31(5) (2011)
- [10] Coulouris, G., Dollimore, J., Kindberg, T., Blair, G.: *Distributed Systems - Concepts and Design*, 5th edn. Addison-Wesley (2011)
- [11] Korkhov, V.V., Moscicki, J.T., Krzhizhanovskaya, V.V.: The User-Level Scheduling of Divisible Load Parallel Applications With Resource Selection and Adaptive Workload Balancing on the Grid. *IEEE Systems Journal* 3(1) (March 2009)

Software Support of the Risk Reduction Assessment in the ValueSec Project Flood Use Case

Jacek Bagiński

Institute of Innovative Technologies EMAG,
40-189 Katowice, Leopolda 31, Poland
jbaginski@emag.pl

Abstract. The chapter presents information about the first stage of validation of the OSCAD tool for the risk reduction assessment within the decision support process. First, general information about risk management and risk assessment is given, and relations of the risk assessment with the flood issue are described. Basic information about the ValueSec project and its relations with risk assessment is presented. Next, the results of first experiments heading for OSCAD usage as one of the possible elements supporting the Risk Reduction Assessment (RRA) software pillar in the ValueSec project are described. The possibility of OSCAD usage for the RRA pillar was validated on the example of the so-called “flood use case” of the ValueSec project. This use case relates to the assessment and selection of flood countermeasures. The main objective of the validation is to find out if the risk assessment method implemented in OSCAD can be used for the flood issue.

1 Introduction

The risk management process and risk analysis related issues are nowadays quite well defined and established in standards. The ISO 31000 [1] and ISO 31010 [2] standards can be mentioned here which distinguish, among others, the risk assessment and risk treatment activities within the whole risk management process.

The same approach is proposed by ISO 27005 [3] which contains recommendations for the information security risk management.

The risk analysis and risk assessment aspects, as well as the risk level reduction are also elements of many other normative documents, standards and recommendations, for example standards for occupational health and safety management systems (OHSAS 18001 [4]) which relate also to such issues as threats identification, risk assessment and control.

Regardless of the application domain of the risk management and risk assessment, the basic definition of risk level is mostly described in the same way, as a combination of incident (undesirable event) consequences and the likelihood (probability) of its appearance:

$$R = C * L \quad (1)$$

This formula (1) can be extended with additional elements, depending on the analysis object and the results purposes. It can be supplemented for example with financial aspects or other parameters influencing the risk value, like security measures which reduce the risk level, their effectiveness, quality, etc.

The same approach is also used in the floodplain management programs, which must consider the risk management aspect.

The requirements on the risk assessment in the flood domain come out of the EU directive 2007/60/EC [5]. Due to the Directive requirements, the EU member countries started during the last few years to launch flood management programs with risk assessment as one of key elements. For example, the Scottish Environment Protection Agency (SEPA) [6] began The National Flood Risk Assessment program. Based on this program results, the document *Flood Risk Management Planning in Scotland: Arrangements for 2012 -2016* [7] was prepared.

Other countries started to implement the risk management process and risk assessment activities within the flood management programs. For example, on the 30th of July 2010, in Poland, the ISOK project began (“Informatyczny system osłony kraju przed nadzwyczajnymi zagrożeniami”) [8], whose result was a report [9] with the preliminary assessment of flood risk in Poland.

An important element of the risk management process is the security measures (in case of flood – flood countermeasures) selection. The selection of appropriate security measures is a vital problem for the decision makers, who have to take efforts to keep their activities and decision making process transparent. Their decisions must be backed by evidences of relevance selected solutions. These evidences, in the case of security measures selection, may contain also risk analysis results about the estimated efficiency, effectiveness, reliability, economic factors, and other costs and benefits (like social, environmental, etc.).

The usage of software support for the security measures selection provides such transparency and possibility to define clearly risk assessment rules and to document the whole process. It also confirms that all necessary activities for the best variant selection were performed.

The implementation of a software tool for such support is one of the main goals of the ValueSec project. Its assumptions were described in more detail in [10] and on the project website [11]. Within the project there will be a tool developed which will support the decision making process. One of the test cases is the above mentioned process of security measures selection in case of a flood.

2 RRA in the Three Pillars Context

According to the assumptions worked out in the course of the project, while selecting the security measures it is necessary to take into account such factors as the risk reduction level offered by a security measure or a group of security measures, the relation between costs and benefits gained from the measure implementation, and the so called “qualitative criteria”, i.e. the assessment of “unmeasurable” parameters and impacts, such as social satisfaction, cultural, political, environmental aspects, etc. [11]. Therefore it was assumed the software supporting the security measures selection, developed within the project, should be based on three pillars:

- Risk reduction analysis (RRA),
- Cost-Benefit Analysis (CBA),
- Qualitative Criteria Analysis (QCA).

Within the project a tool will be developed that will support the operations of these three pillars. The validation of this tool will be performed in the five preliminarily chosen areas (contexts): public mass events, mass transportation, airport security, communal security (with a flood protection application scenario), cyber security

Due to the fact that the risk reduction analysis can be made using a number of risk analysis tools, it was assumed that for the implementation of the first pillar the existing tools will be used, while the CBA and QCA pillars will be fully implemented within the project. For the implementation of the RRA pillar, the following software tools were considered: Riger from ATOS, Lancelot from WCK, RAS from Fraunhofer Institute, OSCAD from EMAG.

The tools were assigned to particular application scenarios in each context with a view to check the correctness of the adopted solution about using a ready-to-use risk analysis tool. OSCAD was selected to fulfill the RRA pillar in the “Flood use case”. “Use case” can be defined as the set of security measures, which contains the measures for evaluation in the ValueSec toolset.

3 Possibility of OSCAD Usage in RRA Pillar

The OSCAD system was developed at the Institute of Innovative Technologies EMAG within a project co-financed by the National Centre for Research and Development (NCBiR) [12]. The OSCAD software was developed to support the integrated system, consisting of a business continuity management (BCM) and information security management (ISM) system. Risk analysis is an important element of both these management systems.

The risk analyzer module implemented in OSCAD was worked out mainly based on the requirements and recommendations of such standards as BS 25999 [13], ISO 27001 [14] and ISO 27005 [3]. They describe requirements and recommendations for BCMS and ISMS. The possibilities of the OSCAD system and the

risk analysis methods have already been described in [15], [16], [17]. The adopted solution is partly based on the approach described in [18]. The adopted method of threats and vulnerabilities assessment corresponds to earlier works conducted in this domain in the EMAG Institute and described in [19], [20]. However, the method was modified and simplified.

The security attributes (confidentiality, integrity and availability) assessment was distinguished in the form of the Business Impact Analysis – BIA. The low level analysis in turn [18], called detailed analysis in OSCAD, does not take into account a part of parameters proposed in [18] and related to risk assessment. Thanks to that, the method used in OSCAD is simpler while the analysis can be performed quickly and without any program support (if there is no such possibility). Additionally, there is no analysis of some economic parameters (e.g. cost efficiency of security measures or return on investment). With OSCAD used for RRA in ValueSec, the lack of the economic parameters analysis is not an obstacle since this issue will be taken care of by the CBA pillar.

For the OSCAD tool a number of experiments and validations were performed, e.g. for the pharmaceutical business (delivery of medications, medical supplies) and the mining sector [21], but these experiments were not related to the flood domain and were oriented on a wider scope of security management than just risk management.

According to the EU directive [5] on the assessment and management of flood risks, for this type of risk management the Member States shall establish objectives, which will focus on the reduction of flood consequences (for human health, the environment, cultural heritage, economic activity) and/or reduction of flood likelihood. According to another directive requirement, during the risk analysis the existing security measures should be considered. This means that, the way of calculation used for the risk assessment in the OSCAD tool can be easily adapted to the risk assessment in the flood domain.

The OSCAD method uses the following formula for the risk calculation:

$$R = \frac{C * L}{SMi * SMta} * Pc, \quad (2)$$

where R means risk value, C and L mean Consequences and Likelihood of incident (flood) appearance. SMi and $SMta$ parameters relate to existing or planned security measures. Pc parameter means value of process criticality, which can be assessed during the business impact analysis.

SMi and $SMta$ were initially intended as security measures implementation level and technical advancement level values. But thanks to the possibility of free definition of values and descriptions of these parameters, they can be adapted to the flood domain needs. Such a possibility is provided by the mechanism of dictionaries implemented in OSCAD (Fig. 1). Based on the [9] $SMta$ and SMi parameters can be defined for example as a class of the flood countermeasure structure, and the safety status of the flood countermeasure structure.

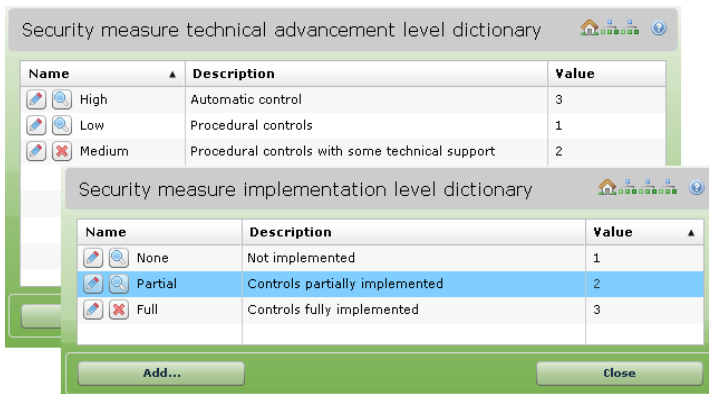


Fig. 1 Example of security measures assessment scales definition in OSCAD dictionaries

The same mechanism of dictionaries can be used to define and describe possible values of consequences (impacts) and likelihood (probability) of an adverse event. Information about required probability levels is also presented in the EU directive [5], which distinguishes three possible levels:

- floods with a low probability, or extreme event scenarios;
- floods with a medium probability (likely return period ≥ 100 years);
- floods with a high probability, where appropriate.

Assigning values to these levels in the dictionary will allow to use them in the formula (2) for the risk level calculation in the OSCAD tool. For the validation of OSCAD usage in the flood domain, more precise values of probability (likelihood) were defined (as presented in Fig. 2), but these values can be easily changed and adjusted to the requirements.

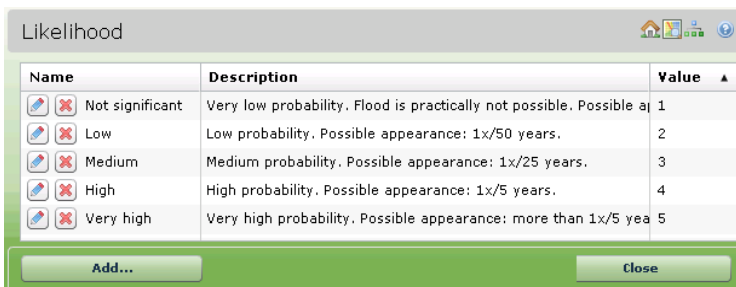


Fig. 2 Example of likelihood assessment scale definition in OSCAD dictionary

In the case of OSCAD usage in the ValueSec frames, it was necessary to analyze the possibility to adapt and configure OSCAD for the needs of the “Flood use case”. For this purpose the documents describing the use case had to be analyzed first. The data were prepared and provided by the project partners (Fraunhofer Institute and The Centre for European Security Studies) with the participation of

the employees of the Saxony-Anhalt Ministry of the Interior and Sports and Saxony-Anhalt Ministry of Agriculture and the Environment.

The initial validation of OSCAD for the analyzed case was conducted based on the data about the flood on the Elbe and Mulde rivers in the Magdeburg area in Saxony-Anhalt. Information about the impact and size of previous floods came from reports about floods in this area. Based on these materials main threats were identified first, along with security measures which were assessed during RRA within the “Flood use case”. Sample values of other parameters required for the analysis in OSCAD were defined in the dictionaries (e.g. assessment scales used to assess the impact, frequency of occurrence, security measures efficiency, etc.).

The dictionaries allow to freely assign values in a defined scale. These can be successive natural numbers (1, 2, 3, ...) or the values can change with different steps (e.g. 1, 5, 25, 100, ...). This way it is possible, to some extent, to control the range of possible output values of the risk level, yet the general shape of the diagram remains the same for the adopted formula (2), as presented in Fig. 3.

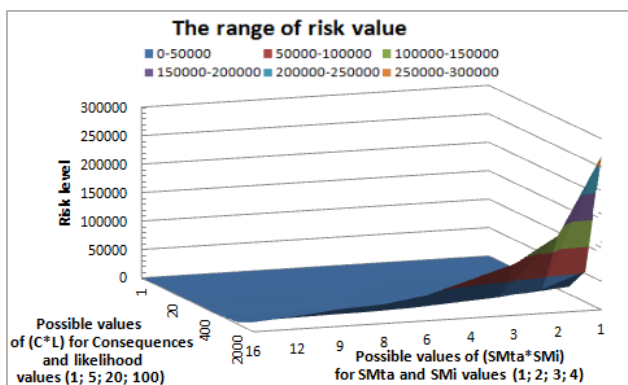


Fig. 3 The example of risk values range depending on the parameters value

After entering the data to the OSCAD data base there was a trial to conduct a risk analysis. The EU directive on floods lists possible aspects (operations) of the flood management process. These operations (processes) are: prevention, protection and preparedness (Fig. 4). That is why there was an attempt to conduct an analysis of processes comprising these operations.

Search for processes

Name: also inactive critical only search in subprocesses

Type: Group:

Number	Name	Type	Group	Owner	Analyses
52	Preparedness	internal		Smith John	BIA, BIA, BCM
50	Prevention	internal		Kowalski Jan	BIA, BCM
51	Protection	internal		Mueller Hans	BIA, ISM

Fig. 4 Example of possible processes for the flood domain registered in OSCAD

The Business Impact Analysis (BIA) allows to assess the criticality (weight) of each performed process, which, as the P_c parameter, is put into the risk formula (2) and affects its value. The more important the process is, the higher is the risk level related to the materialization of the threat in this process. The assessment is conducted based on the definable business loss matrix which defines the considered loss categories, the number of loss levels and the description of each level (Fig. 5). In the course of the BIA analysis it is necessary to assign to each assessed process the estimated level of losses which can occur after losing the security attribute.

Business loss category	Level1 [Short time]	Level2 [Medium time]	Level3 [Long time]
Environmental	No impacts or not significant impacts (recovery possible within 1 year).	Medium impacts (recovery possible within 5 years)	High impacts (recovery possible within more than 5 years)
Financial loss	Below 10.000.000 €	10.000.000 € - 1.000.000.000 €	More than 1.000.000.000 €
Loss of live an i...	No or minor injuries.	Several injured	At least one fatality.
Political	No or not significant	Possible short-lived	Possible claims of other

Business loss category	Level	Justification
Environmental	2	Wrong, incomplete disaster recovery plans for communication ro
Financial loss	1	Loss of information integrity related to communication infrastru
Loss of live an injuries	3	Loss of integrity of information required for communication infras
Political	1	Consequences: At least one fatality.
Social dissatisfaction	1	Loss of integrity of information about traffic, detours, timetables,

Fig. 5 Example of business loss matrix used for the assessment of possible losses in BIA

If there are no processes distinguished for the “Flood use case”, and the flood risk management is treated as one process, it is possible to have value ‘1’ for the assessment of the impact of security attributes loss. Then the value of the process will be equal to ‘1’ and it will be possible to proceed to the detailed analysis.

Within this analysis threats and vulnerabilities (i.e. weak points which can contribute to the threat materialization) are identified (Fig. 6). The existing security measures, which reduce vulnerabilities and counter threats, are also identified.

Threat/Vulnerability	Probab	Impact	SM adv	SM imp	Risk (target/prese)	SM cost	The level of risk a
Damage of a dam or dike:					12 (108)	340000 (10000)	12 (108)
Faults in personnel training	2 (3)	4 (4)	2 (1)	3 (2)	12 (54)	60000 (10000)	8 % (38 %)
Lack of or inappropriate rev	2 (3)	4 (4)	3 (1)	3 (1)	8 (108)	280000 (0)	6 % (75 %)
Rising water level due to flood					14 (27)	350000 (150000)	14 (27)
Inappropriate monitoring off	3 (3)	3 (4)	2 (2)	3 (2)	14 (27)	250000 (50000)	9 % (19 %)
Lack of drainage or ineffici	3 (4)	3 (3)	3 (3)	3 (2)	9 (18)	100000 (100000)	6 % (13 %)

Fig. 6 The example of threats and vulnerabilities selected in the flood use case

The main threat considered in the case of a flood are the following:

- Heavy rainfall results in the rising water level and flooding of the given area with direct impact (damaged communication and telecommunications infrastructure, loss of lives and health, damages to the natural environment, ...) and indirect impact (social dissatisfaction, disturbances in the functioning of enterprises and public institutions, epidemic threat, ...).
- Breakdown of the early warning system.
- Damage of a dam or dikes (failure to detect a high water level, inappropriate maintenance of flood infrastructure).

OSCAD enables to enter the list of threats, vulnerabilities and security measures to the data base and then to make connections between these values. Threats can have typical vulnerabilities assigned, and the vulnerabilities – typical security measures that reduce them (Fig. 7). The connections make it easier to search for these elements during the risk analysis process.

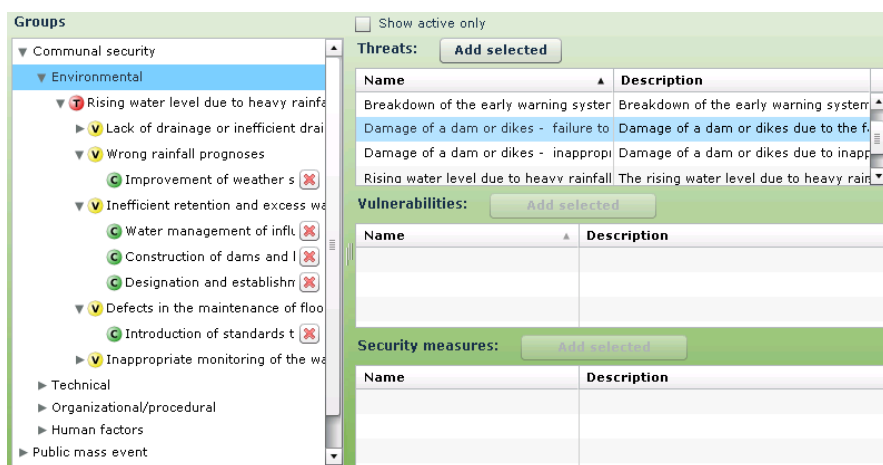


Fig. 7 View of dictionary for Threats-Vulnerabilities-Controls linkage

Next, for these parameters there is an assessment carried out to determine potential impacts of the threat materialization and probability of the event occurrence. Additionally, the functioning security measures are assessed.

Based on the assessments the current risk level is calculated (Fig. 8) which will be a point of reference to the risk values estimated after the security measure implementation. This way it is possible to determine the risk reduction level achieved for security measures considered in a given decision making process. With the “Flood use case” the following security measures are considered:

- ‘Non-measure’, which means leaving the current situation as it.
- Building and/or extension of dikes in the particular area.
- Introduction of standardized crisis management support software for communal crisis management task forces.

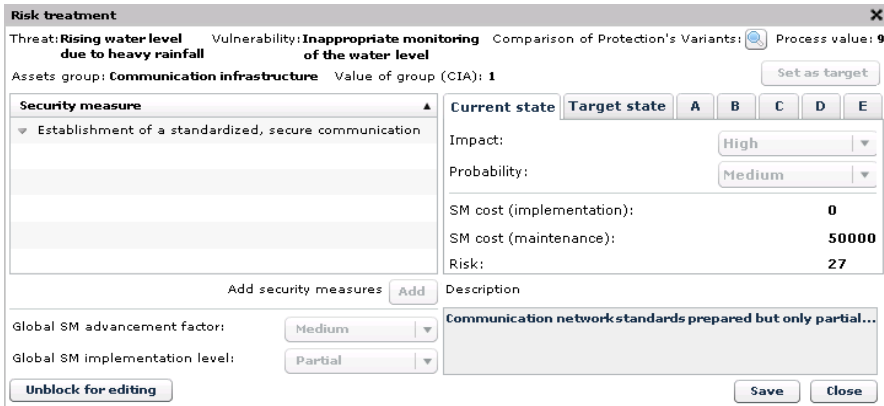


Fig. 8 Risk treatment window – assessment of current state of the risk level

It is possible to compare a few security measures (or sets of measures) in the OSCAD system thanks to the function which allows to assess several variants (up to five, described on tabs marked from A to E – as presented in Fig. 9). For each variant it is necessary to conduct a process of impacts estimation, event probability occurrence and security measures parameters assessment. On this basis, just as for the current risk, the estimated risk level will be calculated.

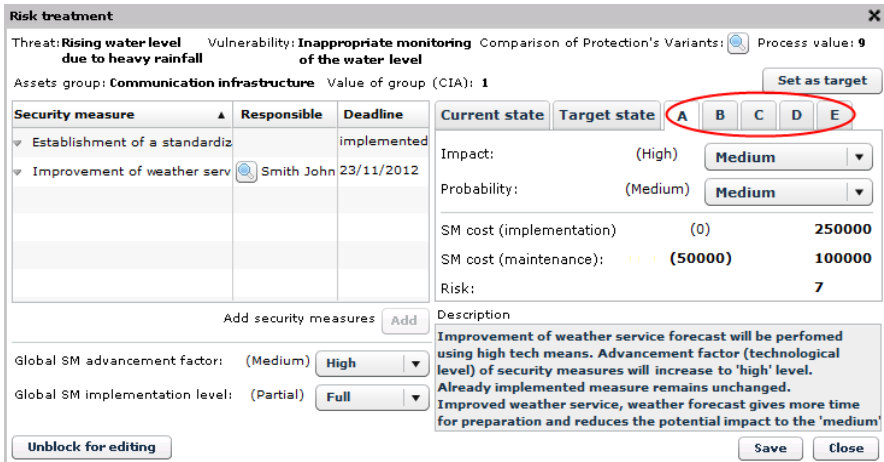


Fig. 9 Assessment of risk parameters for security measures ‘A’ variant

As it can be seen in Fig. 8, the current risk level for the analyzed case was equal to 27. While for the variant A (Fig. 9) the risk value was calculated at the level 6,75. Results for three example variants are presented in the Table 1. While comparing the analyzed variants it can be seen that the best risk reduction can be achieved with variant C, but the cost is very high. A similar risk reduction level can be achieved by implementing the A variant with moderate costs.

Table 1 Comparison of risk assessment parameters for analyzed variants

Risk assessment iteration	Risk level	Overall cost
Current risk	27	50000
A variant	6,75	350000
B variant	13,5	250000
C variant	6	1580000

As it was assumed for the ValueSec project tool architecture, the result of this stage of analysis (RRA pillar) should provide simple, clear information about the risk reduction level for each considered security measures set to support the decision maker in selecting the most appropriate variant of measures. The OSCAD tool gives at the end of the detailed risk analysis several different data sets related to costs, the risk level, and the level of impacts and probability estimated after the security measures implementation. Using the data gathered in the OSCAD tool, information can be processed manually and presented, for example, in the form of tables (as presented in Table 1). But the tool offers also a quick view window with predefined charts (Fig. 10) for presenting values of each pair threat-vulnerability, existing security measures and variants of planned security measures.

These charts present information related to the level of reduction of each risk parameter, such as expected risk level for each variant, or estimated value of such parameters (as impact, likelihood, levels of security measures parameters).

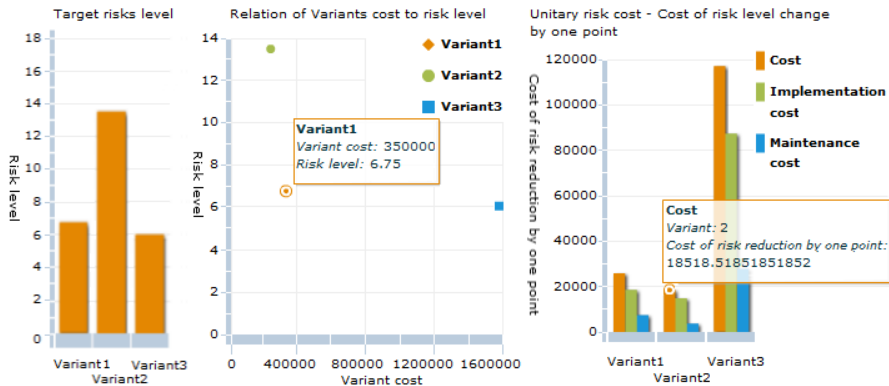


Fig. 10 The example of charts presenting results for each variant

By comparing these values with those calculated for the current situation, one can get information about the level of reduction in the case of a particular variant implementation. Other charts present information from the financial point of view. This is information about the risk level that can be achieved by the implementation of each set of measures with relation to their cost, and information about unitary cost of risk which means the cost of changing the risk by one-point.

The latter value (unitary cost of risk) is calculated in OSCAD with the use of the following formula (based on the solution proposed in [18], [20]):

$$UC = \frac{C_{i+1}}{RV_i - RV_{i+1}}, \quad (3)$$

where UC is the Unitary cost of risk, C_{i+1} means the cost of new security measures, RV_i means Risk Value for the i -th analysis and RV_{i+1} is the estimated value of risk after the new security measures implementation.

The information presented on the charts, received from the OSCAD database, can be next sent to the main ValueSec tool as the result of the RRA pillar, where together with Cost-Benefit Analysis and Qualitative Risk Assessment results will be presented to the decision makers as the result of the decision problem analysis.

4 Conclusions

The results of the first attempts to use the OSCAD software for the risk reduction analysis within the decision-support process were presented to the project partners. They seem to confirm the assumption that the general method adopted in the software, taking into account extra parameters (such as applied security measures and their efficiency or the implementation level), can be applied also in the selection of anti-flood security measures.

The results of the analysis conducted in the RRA pillar can be then considered during the analysis in the successive pillars (CBA and QCA). This way, during the whole cycle of the analysis and support of the decision about the security measures selection there will be certain aspects considered, such as the risk reduction level of particular variants, economic efficiency analyzed in CBA. Possible impacts of the security measures on the factors which are difficult to measure, such as social, political, etc. are also taken into consideration during the QRA.

One of the crucial issues to solve in the ValueSec project is the scope and form of input data and results, as well as the way of exchanging these data between particular tools selected for the risk assessment (Riger, Lancelot, OSCAD, RAS) within this RRA pillar. Data exchange between pillar is also one of the implementation challenges in this project.

One of the tested methods of data exchange was the usage of a broker which communicates with the database of each tool. The broker gets information about threats, vulnerabilities, security measures, or any other required element saved in the database. The user sends queries in the SPARQL query language to the broker which returns results prepared based on the mapping file. This file serves the mapping between objects in databases on classes and properties defined in the ontology (more information about the ontology usage in the security domain can be found in [22], [23], [24]). Such approach requires information about the part of the database structure which stores data for the exchange. While there are some issues regarding the access to the database and presenting the database structure, this approach was tested only with the different instances of OSCAD systems. Access to the database of each tool (Fig. 11) was simulated by connection with different instances of the OSCAD tool.

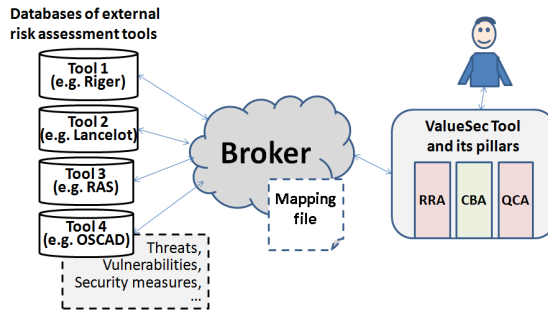


Fig. 11 A simple diagram of tested method of data exchange between tools in RRA

Due to the fact that EMAG is the owner of OSCAD and has access to source codes, it is possible to make some changes in the software for the needs of ValueSec if it is necessary to make extra lists or give access to extra output data. The software itself, in the majority of its modules, offers a function of data export to the CVS format which can be then read by any calculation sheet (e.g. Microsoft Excel) while the exported data can be used to work out a list or diagram.

Some parts of the OSCAD system were also verified against the Common Criteria standard. As a result the security architecture of the tool was assessed according to possible vulnerabilities which could be exploited by threat agents. Consequently, the tool becomes more reliable to users. The OSCAD system can be placed at remote sites (as it was mentioned in the description of the data exchange test). In that case the assurance of the system can be strengthened by applying another Common Criteria approach called Site Certification, as it was described in [25].

Currently, when the article is being written, further tests are underway. It is planned to have meetings of the project partners and representatives of Saxony-Anhalt ministries to specify input data and configure the software for the case of flood risk assessment of the Elbe river in Saxony-Anhalt.

References

- [1] ISO 31000:2009 – Risk management - Principles and guidelines
- [2] ISO/IEC 31010:2009 – Risk management - Risk assessment techniques
- [3] ISO/IEC 27005:2008 – Information technology - Security techniques - Information security risk management
- [4] BS OHSAS 18001:2007 – British Standard for occupational health and safety management systems – Requirements
- [5] Directive 2007/60/EC of the European Parliament and of the Council of 23 October 2007 on the assessment and management of flood risks
- [6] <http://www.sepa.org.uk>
- [7] Flood Risk Management Planning in Scotland: Arrangements for 2012 - 2016, http://www.sepa.org.uk/flooding/flood_risk_management/national_flood_risk_assessment.aspx

- [8] isok.imgw.pl (accessed January 18, 2013)
- [9] Raport z wykonania wstępnej oceny ryzyka powodziowego. IMGW PIB. W konsultacji z Krajowym Zarządem Gospodarki Wodnej (2011), http://www.kzgw.gov.pl/files/file/Materialy_i_Informacje/WORP/Raport.pdf (accessed January 18, 2013)
- [10] Białas, A.: Risk assessment aspects in mastering the value function of security measures. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *New Results in Dependability & Comput. Syst. AISC*, vol. 224, pp. 25–39. Springer, Heidelberg (2013)
- [11] ValueSec Project, <http://www.valuesec.eu> (accessed January 10, 2012)
- [12] Institute EMAG, Reports of a specific-targeted project “Computer-supported business continuity management system – OSCAD” (2010-2012)
- [13] BS 25999-2:2007 Business Continuity Management – Specification for Business Continuity Management
- [14] ISO/IEC 27001:2005 – Information technology – Security techniques – Information security management systems – Requirements
- [15] Bagiński, J., Rostański, M.: The modeling of Business Impact Analysis for the loss of integrity, confidentiality and availability in business processes and data. *Theoretical and Applied Informatics* 23(1), 73–82 (2011) ISSN 1896-5334
- [16] Baginski, J., Białas, A.: Validation of the software supporting information security and business continuity management processes. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability. AISC*, vol. 170, pp. 1–17. Springer, Heidelberg (2012)
- [17] Białas, A.: Computer support in business continuity and information security management. In: Kapczyński, A., Tkacz, E., Rostanski, M. (eds.) *Internet - Technical Developments and Applications 2. AISC*, vol. 118, pp. 155–169. Springer, Heidelberg (2012)
- [18] Białas, A.: *Bezpieczeństwo informacji i usług w nowoczesnej instytucji i firmie*. WNT Publishing House, Warsaw (2006)
- [19] Białas, A.: Development of an Integrated, Risk-Based Platform for Information and E-Services Security. In: Górski, J. (ed.) *SAFECOMP 2006. LNCS*, vol. 4166, pp. 316–329. Springer, Heidelberg (2006)
- [20] Białas, A., Lisek, K.: Integrated, business-oriented, two stage risk analysis. *Journal of Information Assurance and Security* 2(3) (September 2007) ISSN 1554-10
- [21] Białas, A., Cała, D., Napierała, J.: Wspomaganie zarządzania ciągłością działania zakładu górniczego za pomocą system OSCAD. *Mechanizacja i Automatyzacja Górnictwa, Czasopismo Naukowo – Techniczne* 7(497), 11–25 (2012)
- [22] Białas, A.: Security Trade-off – Ontological Approach. In: Akbar Hussain, D.M. (ed.) *Advances in Computer Science and IT*, pp. 39–64. In-Tech, Vienna-Austria (2009) ISBN 978-953-7619-51-0, <http://sciendo.com/articles/show/title/security-trade-off-ontological-approach?PHPSESSID=kk15c72nt1g3qc4t98de5shhc2>
- [23] Białas, A.: Ontological Approach to the Business Continuity Management System Development. In: Arabnia, H., Daimi, K., Grimaila, M.R., Markowsky, G. (eds.) *Proceedings of the 2010 International Conference on Security and Management, The World Congress In Applied Computing – SAM 2010, Las Vegas, USA, July 12-15, vol. II*, pp. 386–392. CSREA Press (2010) ISBN: 1-60132-159-7, 1-60132-162-7 (1-60132-163-5)

- [24] Bialas, A.: Common Criteria Related Security Design Patterns for Intelligent Sensors—Knowledge Engineering-Based Implementation. *Sensors* 11, 8085–8114 (2011), <http://www.mdpi.com/1424-8220/11/8/8085/>
- [25] Rogowski, D., Nowak, P.: Pattern based support for site certification. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability*. AISC, vol. 170, pp. 179–193. Springer, Heidelberg (2012)

Risk Assessment Aspects in Mastering the Value Function of Security Measures

Andrzej Białas

Institute of Innovative Technologies EMAG,
40-189 Katowice, Leopolda 31, Poland
a.bialas@emag.pl

Abstract. The chapter presents the risk management approach applied in the EC FP7 ValueSec project. The security measures selection process is based on three pillars: Risk Reduction Assessment (RRA), Cost-Benefit-Analysis (CBA) and Qualitative Criteria Assessment (QCA). The ValueSec tool set, which is elaborated in the project, should be equipped with components corresponding to these pillars. The chapter overviews the researches of the project focused on the decision model elaboration and selection of existing method to be implemented, or existing tools to be integrated in the ValueSec framework. Risk management is a broad issue, especially in five of the project assumed contexts. For this reason more specialized components are allowed for the RRA pillar. Currently the project passes to the implementation and use case experimentation phase. The chapter shows the general architecture, currently implemented and the RRA component example.

1 Introduction

The chapter concerns the selected aspects of risk management with respect to the ValueSec project [1] financed by the EC 7th Framework Programme.

The objective of the project is to improve the decision process related to security measures selection so that the proposed measures could take into consideration the stakeholders' needs and interests to the highest possible extent. Along with the developed decision support methodology, a software tool is prepared for policy level stakeholders in the field of security. The project results should allow to support policy decision makers in making better informed decisions.

The scientific problem to be solved in the interdisciplinary ValueSec project can be defined in the following way: developing a computer-aided decision support methodology in security concerning the selection of security measures, so that the measures not only properly affected the risk but also were cost-effective and took into account social, political and legal restrictions which are related to the decision making process. Taking into account these restrictions, here called qualitative factors (criteria), is the basic added value of the project.

The project passed its half-way point. The key analytical works were completed, the assumptions for the decision support methodology were made [2], [3], [4], [5], methods or tools for implementation were selected [6], [7], [8], [9], the functional design and architecture of the software were developed, and the validation process of the developed solution is being prepared.

The chapter will focus on one of the project pillars which is risk assessment, therefore in the next section the ValueSec approach to the security-related decision support will be presented.

Section 3 will feature the analysis of the existing methods and tools with a view to select some of them for implementation in the ValueSec project. Sections 4 and 5 will feature the general architecture and an example of a risk assessment tool implementation. The final summary will present operations that are planned in the course of the project and related to risk assessment in the project contexts.

2 ValueSec Approach to Security Related Decisions

The decision making process related to security measures is important to many organizations, projects, social groups, and individuals as it affects security, business efficiency and social acceptance for these security measures. At the same time it is an extremely complex process since it has to take into account a number of factors of different, complicated and still unexplored nature.

Generally speaking, security measures are applied to reduce risk. Among many security measures that are able to reduce risk in a particular case, a part may be economically inefficient or impossible to apply due to objective restrictions. The decision about the measure selection is a multi-dimensional issue.

The objective of the ValueSec project is to master the value function of security measures [10]. This value function is researched, all its arguments (factors) and their multidimensional impact to the security problem are identified.

The decision about selecting a security measure in a given situation should be worked out on the basis of the following three factor groups, each including a number of detailed issues:

- the security measure should be able to affect the risk volume sufficiently (based on the risk appetite) in order to provide security on a suitable level,
- the security should be cost-effective in order not to reduce the efficiency of operations and not to incur unnecessary costs,
- the security should take into account a number of restrictions: social, psychological, political, legal, ethical, economical, technical, environmental, etc. – in order to use the security measures in practice; in the project terminology these factors are called qualitative criteria.

Each group of issues in the project is called a pillar, thus one can distinguish three pillars in ValueSec:

- Risk Reduction Assessment (RRA) pillar,
- Cost-Benefit-Analysis (CBA) pillar,
- Qualitative Criteria Assessment (QCA) pillar.

The first pillar concerns a vast domain of risk assessment, including risk analysis and risk evaluation. The risk analysis is conducted in many domains, including widely understood security. Different strategies, methods and tools are used, models of different degrees of detail are applied. They provide a set of factors which are taken into account in the decision making process concerning the security measures selection [11]. This pillar is responsible for calculating risk reduction resulting from the application of the given security measure. Due to the existence of many theories, methods and tools for conducting and supporting risk assessment, an exhaustive two-stage analysis of these TMTs had to be conducted to select those that could be implemented or integrated in the ValueSec framework. This issue will be discussed in detail in section 3.

Out of the security measures that affect properly the risk level it is necessary to choose cost-effective variants which comply with different restrictions characteristic of a concrete situation. The monetary approach is used. The key issue is an economic analysis about the cost-efficiency of the applied measures with respect to their costs and benefits. The applied economic models provide the second set of factors considered in the decision making process. This pillar is responsible for calculating, in monetary units, negative and positive effects of applying a certain measure [12].

The objective of the CBA analysis is to assess from the economic point of view the impact of security-related decisions. This analysis encompasses the following categories:

- costs of provision and investment,
- direct and indirect maintenance costs,
- immaterial costs,
- direct and indirect benefits.

Each of these categories has its subcategories. For example, the category of direct and indirect costs has the following subcategories:

- operational costs, comprising, e.g. costs of equipment and its modifications, costs of personnel training,
- maintenance costs, comprising the costs of planned and unexpected reviews and repairs of equipment, costs of spare parts, costs of IT services and maintenance,
- costs of utilization, comprising, among others, costs of a system closure, its disassembly, costs of recycling.

The category of direct and indirect benefits, in turn, includes the following subcategories:

- direct economic benefits, such as the estimated sales volume or incomes from the sale of patents and licenses,
- benefits resulting from the reduction of risk and the degree of vulnerabilities to threats,
- social, legal and political benefits, e.g. better image of the organization, higher consumption, acquiring new clients, better contacts with the organization's business and legal environment.

The third pillar comprises the analysis of restrictions with the use of varied factors which are difficult to determine [8]. This is a new research issue. About 120 issues in several groups (social, political, legal, etc.) were identified and the character of their relations to security measures was determined. This way the third set of factors was created – qualitative criteria, taken into consideration in the decision making process. This pillar is responsible for the evaluation of other important security-related factors.

The Qualitative Criteria Analysis (QCA) is meant to assess those criteria of the decision making process which cannot be assessed by means of quantitative methods. In this case the assessment process has to take into account a number of immaterial parameters of security-related decision making. These parameters can be assigned to the following groups:

- social parameters (social group level),
- individual parameters (individual level),
- legal regulations,
- social laws and ethics,
- politics,
- economy,
- technologies and science,
- environment.

For example, in the economic group of parameters the following issues are included:

- Does the applied security measure affect the consumption behaviour of the society?
- Does it affect the general investment climate (of the country, region, city)?
- Do the applied security measures affect production processes?
- Can the applied security measures cause economic losses for an organization, city, region?
- Are the costs of applied security measures proportional to the achieved effects?
- Can the applied security measures increase or reduce the market value of real estate (in a city, region)?

Though the above issues have an economic background, they cannot be expressed in monetary units as it was the case with categories used in the CBA analysis.

In the ValueSec project the above three groups of issues are considered in five application domains, called contexts [13]: public mass event, public mass transportation, air transportation/airport security, communal security planning, cyber threat.

In the course of the ValueSec project fulfillment the ValueSec framework is developed. It integrates a set of tools corresponding to the three pillars. The tools are to provide a set of analytical data to be used in the security-related decision-making process. Security-related decisions are made based on the conceptual decision model [3], prepared for ValueSec, presented in Fig. 1.

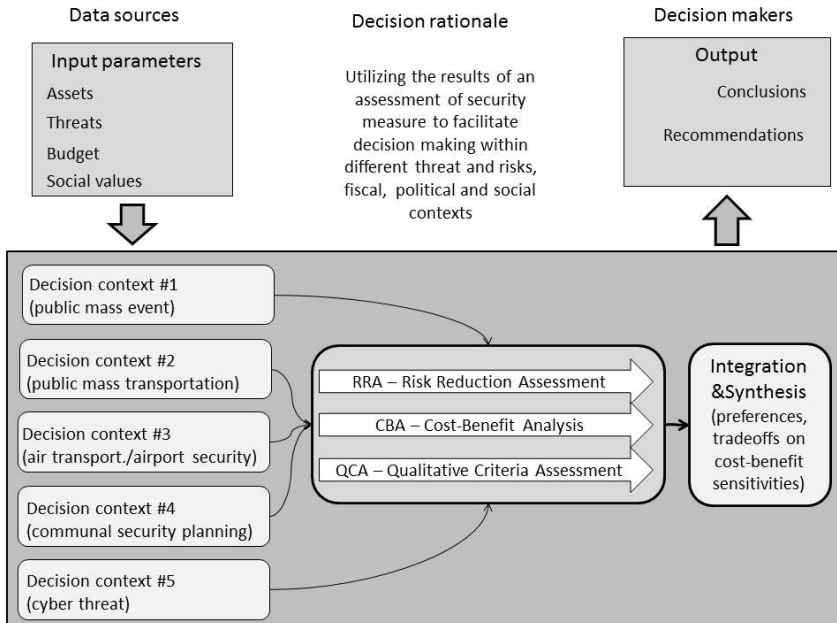


Fig. 1 Conceptual decision model

For the given decision context the decisions have three phases:

- acquisition of input data and definition of data sources that will be the base for the analysis of variants; the data encompass assets, threats, vulnerabilities, budget and time restrictions, soft factors;
- evaluation of different measures – as decisions variants; tools for three pillars are used (RRA, CBA and QCA); different experiments are provided with variants to produce information for decision makers,
- integration and synthesis; information obtained from the previous step are analyzed, integrated and concluded by decision makers, to produce transparent decisions (considering preferences, restrictions and trade-offs).

3 Components of the ValueSec Framework

The problem was to identify methods which can be implemented as the ValueSec framework components, or tools which can be integrated in this framework – all of them should meet the project needs and restrictions, and indirectly should satisfy the stakeholders' expectations. Before defining the ValueSec framework architecture, 2-stage researches on the current state of technology were performed to identify theories, methods or tools (TMTs) which can be implemented in the ValueSec framework.

3.1 General Review of the Theories, Methods or Tools

In the first stage the following categories of TMTs were considered:

- Information handling,
- Risk analysis, risk assessment,
- Cost structuring/analysis and evaluation and analysis of societal impacts,
- Structuring and analysis of values,
- Analysis of decision alternatives,
- Other supporting methods, theories and tools,

The following individual assessment criteria (characteristics) were defined for the TMTs assessment framework: name, theory/method/tool, category, summary, objective, functionalities, qualitative/quantitative attributes of interest, weighting of attributes, phase of decision making, type of decision support, type of damage, inclusion of incident probability and uncertainty, assessment of effects, scientific experience, decision-makers' experience, age, complexity, required resources, required competencies, maturity, timeframe, data-related questions, questions about application and implementations, references [6].

The consortium members evaluated 29 TMTs [7]. They were organized in a matrix structure to match the ValueSec requirements:

- expected functionality (3 above mentioned pillars),
- decision-making context (5 above mentioned contexts).

The following 10 methods and tools were transferred to the next stage for further assessment:

- Quantitative Risk Assessment (QRA),
- Risk Measurement /Risk Analysis (RM/RA),
- Expert Choice (EC),
- Lancelot,
- OSCAD,
- Riger,
- Bayesian Network Analysis (BNA),
- Strategic Approaches (SA),

- TableTop Exercise (TTE),
- Magerit.

The theories were excluded due to the restricted project resources for their implementation. For the CBA and QCA pillars no proper solutions were found. For this reason it was decided that they will be implemented by the consortium partners. All methods and tools transferred to the next stage are related to the risk assessment process (first RRA pillar).

3.2 Usability Assessment Criteria and Usability Analysis

The second stage of the methods and tools assessment was focused on the usability and feasibility aspects.

Recommending the right methodology to implement it in the ValueSec tool set, or recommending the given tool to integrate it (as a component) with this tool set, requires the identification of their most favorable features and the elaboration of the usability criteria to assess them. Different types of issues have been considered during the usability assessment criteria elaboration:

- Risk-related, security economics and soft factors (further quality criteria of decisions) methodology; they should cover the requirements and expectations of stakeholders;
- Related to the capability of decision support for policy makers, the urgency of the users' needs, decision context;
- Dealing with the expected efficient use of the required ValueSec project resources and the feasibility within the given time frame and development risks;
- Related to the implementation requirements, methodical innovations, project challenges issues, and constraints of the selected methods.

The usability assessment criteria, elaborated with the use of Microsoft Excel, were specified in [9]. These criteria have a three-level hierarchical structure (Fig. 2). On each level weights were assigned. The assessment results are presented in a tabular and graphical form. There are 8 groups of criteria dealing with:

- The compliance of the considered method/tool with the ValueSec assumptions, objectives – general parameters related to the method/tool compliance, adequacy, and usability with respect to the area determined within the project;
- The parameters regarding the possibility of situation description (framing conditions specification, problem identification);
- The data characteristics parameters concerning the method of description of input/output data, type, source and other data related issues;
- The functional parameters comprising the desirable functions, possible analyses which are performed and supported by the method/tool;
- The recommendations, reports, final decision support related issues, the types of provided reports, the way of results presentation for decision-makers;

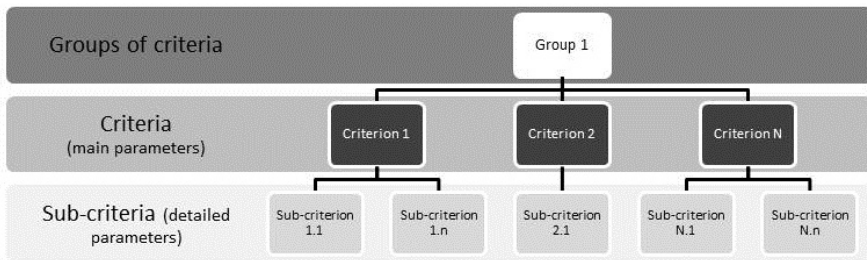


Fig. 2 Hierarchical structure of the usability criteria

- The development/implementation risk parameters concerning the level of risk implied by the use of the given method/tool, its implementation, like source code availability, time, budget and resources required for implementation; such parameters like method/tool age, maturity, general experience, which were identified within WP3, were also taken into consideration; they may have significant influence on the development risk;
- The general challenges specified for the ValueSec project concerning such topics as: scientific challenge, implementation challenge, testing challenge;
- The tool characteristics parameters characterizing mostly the technical aspects of the tool.

Each group contains one or more main parameters (criteria). Additionally, for some main parameters (criteria) there were detailed parameters (sub-criteria) defined. An example of the three-layer structure:

- The “Tool characteristic (technical) attributes” group incorporates;
- The “Software processing (How are the data processed?)” criterion, which includes;
- Several sub-criteria (detailed parameters), e.g.: “online”, “real time”, “batch”, which can be assessed by the given method/tool evaluator.

On each hierarchy level weights can be assigned, allowing to express the importance of the given group in comparison with others, the importance of the given criteria in comparison with others and the importance of the given sub-criteria with respect to others. The initial weight value for each element was set to “1” and it is the lowest possible value. The highest value was not specified a priori, but during the adjustment of criteria weights the maximum value was determined as “3”. The use of the weights will enable to distinguish parameters that are essential for the project and to give better score to particularly desirable characteristics of the methods and tools.

Fig. 3 presents weights for groups of criteria, initially set to “1”. It means all groups have the same importance. For the group “Compliance ...” its four criteria with the same weights are shown.

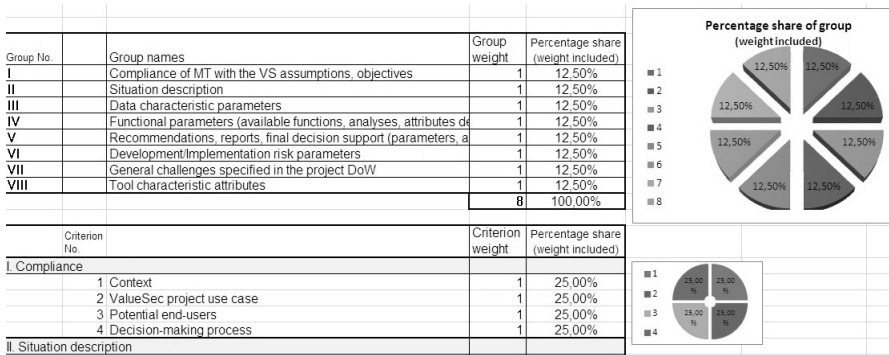


Fig. 3 Weights for groups of criteria and for criteria of the group “Compliance”

Fig. 4 presents a small part of the usability criteria with some data related to one of the evaluated methods/tools called “OSCAD”. Please note: the group “Compliance of considered method/tool with the ValueSec assumptions”, the criterion “context”, and its six sub-criteria: “1a” through “1f”.

				Name		The name of the method/tool (owner)	OSCAD (EMAG)		
				Acronym		The acronym of the method/tool	OSCAD		
				Identifier from document		Link do detailed description placed in	Method/tool categorization		
				Categorization of method/tool			Method/tool categorization		
Source of the parameter	Type of the parameter (General/Tool/Method specific)	Main parameter	Weight factor for main parameter	Detailed parameter (if needed)	Weight factors of parameters	Set of possible values	Assessed value	Value with weight factor included	Description/Justification of assessment
Compliance of MT with the VS assumptions, c					1	Total (for compliance parameters)		#ADR!	
1a	D3.1_v3.0-ch.3 p. 25c	G	Context To which of the ValueSec contexts does the method relate?	public mass event	1	1 - suitable 0 - not suitable	1	1	Oscad can be useful for the risk analysis (threat/vulnerability assessment) for all areas. It is possible to assess possible impacts of the loss of confidentiality/integrity/availability (C/I/A) of a data/process/operation.
1b	D3.1_v3.0-ch.3 p. 25c	G	Main parameter (criterion)	public mass transportation	1	1 - suitable 0 - not suitable	1	1	Matrix with categories of expected losses and description of each possible level of loss must be prepared first to quantify the assessment).
1c	D3.1_v3.0-ch.3 p. 25c	G		air transportation/airport security	1	1 - suitable 0 - not suitable	1	1	
1d	D3.1_v3.0-ch.3 p. 25c	G		communal security planning	1	1 - suitable 0 - not suitable	1	1	
1e	D3.1_v3.0-ch.3 p. 25c	G		cyber threat	1	1 - suitable 0 - not suitable	1	1	
1f	D3.1_v3.0-ch.3 p. 25c	G		other outside the ValueSec contexts - specify in description	1	1 - suitable 0 - not suitable	1	1	
				Detailed parameters (sub-criteria, elements)					

Fig. 4 Hierarchical structure of the usability criteria

These issues express whether “OSCAD” is suitable or not for particular contexts of the ValueSec project.

All sub-criteria have global meaning, which is expressed by “G”. Please note weights assigned on each of the three levels of the usability criteria. During the method/tool assessment two columns “Assessed value” and “Description/justification of assessment” are filled in with data. For each assessed method/tool about 200 such detailed items should be filled in by scores.

The assessment against the methods/tools feasibility and the rationale of further implementation were based on the elaborated usability criteria. The functions and properties offered by the particular method or tool were analyzed. Particular attention was paid to how the method or tool fulfills the needs and requirements of the ValueSec project.

The project partners made an overview of the evaluation process of the 10 pre-selected, above mentioned methods/tools with the use of usability criteria. The assessment process encompasses two steps:

- Reviewing the particular methods/tools with respect to the usability criteria and their groups (horizontally, by methods/tools) – example Fig. 5;
- Identifying strengths and weaknesses of the preselected methods/tools with respect to the usability criteria (vertically, each method/tool against criteria) – example Fig. 6.

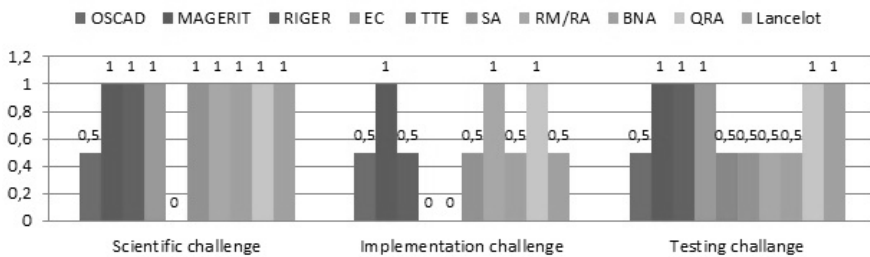


Fig. 5 General challenges specified in the project description of work – results

The results obtained during the assessment process are considered as “sensitive data”. For this reason the deliverable D4.1 Part 2 has a document status RE (restricted), and below only partial results are presented as examples.

Fig. 5 presents the results related to the criteria group “General challenges specified in the project DoW (Description of Work)” as the example. For all methods/tools selected for the implementation of the ValueSec tool set, it will be a challenge to combine them together, to get as a result one comprehensive, common tool supporting the decision making process. It will be a challenge to integrate different risk analysis tools, and furthermore supplementing this tool set with other analyses (qualitative criteria as well as cost-benefit analysis).

Some of the considered tools have already been implemented (Lancelot, Riger, OSCAD) in an IT environment. They are used with respect to threats and security measures in the IT domain. Other selected and assessed tools come from other business sectors. Testing the risk analysis process in different environments (for different threats, vulnerabilities, and security measures) can be a challenge.

Fig. 6 presents another example – the results of the OSCAD assessment. They can be shown because OSCAD is developed by the author’s organization.

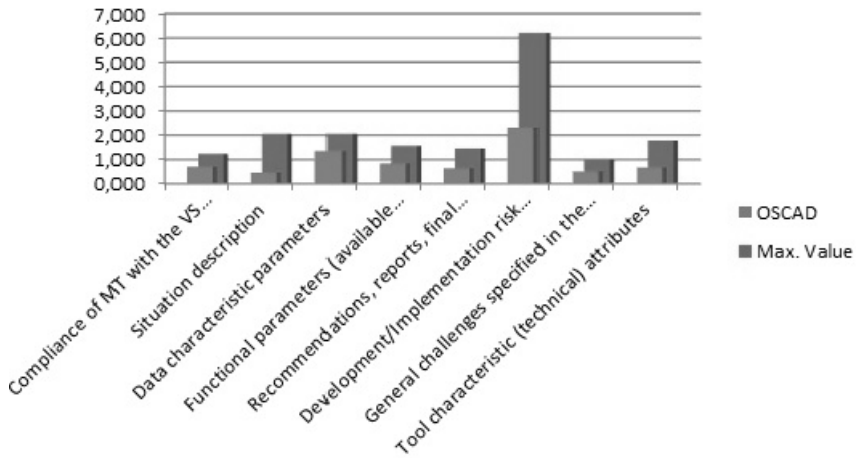


Fig. 6 OSCAD assessment – results for each group of criteria

The OSCAD tool “Data characteristic parameters” received a high score. The entered data are quantified which enables their computer-based analysis (risk level calculation). Similarly to Lancelot, “Situation description” covers, first of all, information about existing threats and vulnerabilities. There is no description of other aspects, such as time for conducting analyses, decision making dimension and decision making domain, assumed budget and time for security measures implementation, or expected benefits.

The tool has a very wide range of functions supporting information security and business continuity management. However, in terms of the ValueSec project requirements it enables, first of all, to conduct a risk analysis for the identified threats and vulnerabilities. Still, it does not offer other analyses, such as the “Cost-Benefit analysis”. “Soft criteria” (now called “Qualitative criteria”) are considered with respect to losses incurred due to the loss of confidentiality, integrity or availability of business processes. Their use in the implementation of the ValueSec tool set would require changes in the software. Hence the overall score of functional parameters is on a medium level.

Low score was given to the ability to generate reports. The tool does not offer too many possibilities of reports adaptation. The only way to generate reports other than the predefined ones is to export data from the selected views to the csv format and then process the data in a calculation sheet, e.g. in MS Excel.

Low score in the range of risk-implementation parameters results from the fact that OSCAD is a new tool. The general knowledge about the tool is not widely available. It is necessary to make some changes in the application and the owner’s resources are limited. Nevertheless, it is possible to make changes and adaptations.

The source code belongs to a member of the ValueSec consortium, which is certainly a great advantage.

Technical parameters were scored on a medium level. Technical documentation and user’s documentation need to be supplemented. There are limited possibilities as far as the configuration changes for the ValueSec tool set implementation, scalability and interoperability are concerned.

During the assessment process the most favorable methods/tools from the technical point of view and software implementation possibility point of view were identified. The results were presented in the form of tables and diagrams on which the assessment value of the given tool is presented against the maximal number of points that can be achieved during the assessment (due to sensitive information related to the method/tools comparison, the D4.1/part 2 deliverable has dissemination status “restricted”).

As a result of the conducted assessment of methods and tools, there were four candidates selected for the final implementation of the RRA pillar:

- Riger, OSCAD – methods focused on assets,
- Lancelot, RAS (QRA) – methods focused on processes.

They will be assigned to concrete application domains, i.e. contexts.

4 ValueSec Framework Architecture

The ValueSec tool set encompasses components of three pillars (Fig. 7).

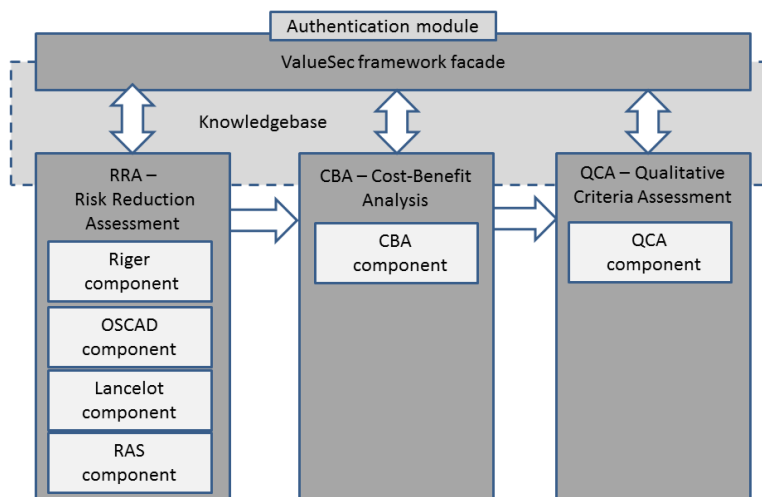


Fig. 7 ValueSec general architecture

For the CBA and QCA pillars the specific components have been developed by the consortium members. For the RRA pillar four different components will be integrated. They have been assigned to given contexts with respect to the context requirements.

Pillars are integrated by a common façade controlling work flow inside the ValueSec tool set. Additionally, some common components (not discussed) exist, such as the knowledge base, authentication module, etc.

5 OSCAD as the RRA Component Example

The OSCAD tool developed by the author’s organization is assigned to the “flood protection use case” of the “communal security planning” decision context. The use case deals with flood prevention measures and will be modeled on the experiences of the German Bundesland Saxony-Anhalt (LSA) during the 2002 flood of the Elbe river.

The OSCAD software system manages business continuity according to the BS25999 standard and information security according to the ISO/IEC 27001 standard [14]. It has several modules, but for the ValueSec RRA pillar the risk management functionality has key importance. For the ValueSec project a dedicated software version was elaborated.

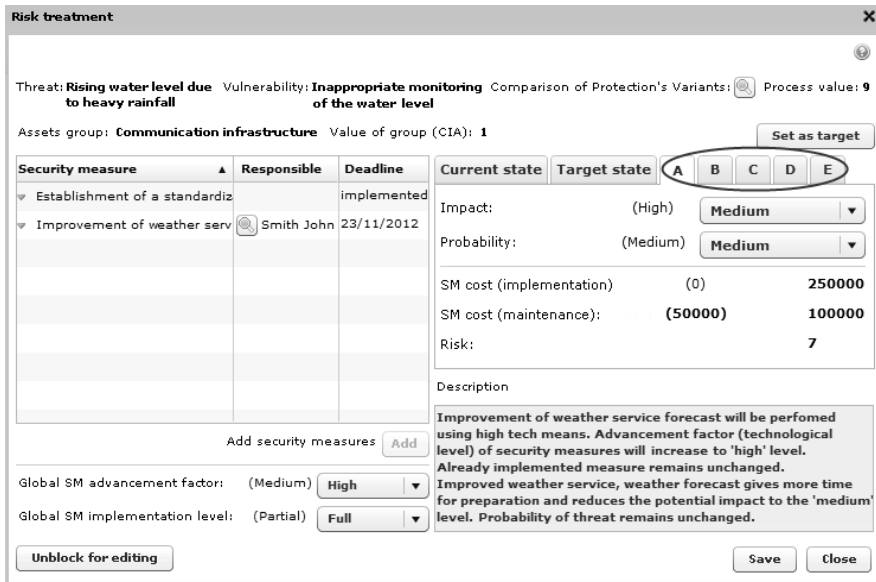


Fig. 8 OSCAD – analyzing variants of security measures (source: EMAG)

Risk management functions are responsible for:

- identification and specification of the business processes, taking into consideration assets related to the particular processes;
- conducting an analysis of harmful influence of losing a continuity attribute on business processes and harmful influence on losing integrity, availability and confidentiality of assets groups related to the given process; this type of analysis is called BIA (Business Impact Analysis) [15] and corresponds to HLRA (High Level Risk Analysis); processes with critical significance for the institution are identified;
- conducting LLRA (Low Level Risk Analysis) which allows to determine the risk value for each triple asset-threat-vulnerability; taking into account the existing security measures, their technical advancement and implementation level;
- selecting security measures which reduce the risk volume; security variants are defined (Fig. 8); the most beneficial variant is considered for implementation, i.e. the one which can reduce the risk and implementation costs the most.

The flood protection use case and the OSCAD tool facilities are discussed in a separate chapter [16].

6 Conclusions

The chapter presents a general approach to risk management applied in the EC FP7 ValueSec project. This approach is focused on mastering the value function of security measures. The ValueSec tool set, which has been elaborated in the course of the project, is based on three pillars: Risk Reduction Assessment (RRA), Cost-Benefit-Analysis (CBA) and Qualitative Criteria Assessment (QCA). The applied measures should be efficient in risk reduction, cost limitation, benefits increase and should be applicable with respect to different restrictions. The project passed its half-way point: analytical works were completed, methods and tools to be implemented as the components were selected, functional design and architecture were defined. Currently the ValueSec tool set is implemented and prepared for use cases experimentations in five decision contexts.

References

- [1] ValueSec web page: <http://www.valuesec.eu> (accessed January 10, 2012)
- [2] D2.1 Decision domains concepts and trends (2011),
<http://www.valuesec.eu/content/d21-decision-domains-concepts-and-trends>
- [3] D2.2 Data model and decision model (2011),
<http://www.valuesec.eu/content/d22-data-model-and-decision-model>

- [4] D2.3 Relational concept between security and politico-economic sphere (2011), <http://www.valuesec.eu/content/d23-relational-concept-between-security-and-politico-economic-sphere>
- [5] D2.5 Report on workshop on user needs and requirements (2011), <http://www.valuesec.eu/content/d25-report-workshop-user-needs-and-requirements>
- [6] D3.1 Framework for the assessment of methods and tools (2011), <http://www.valuesec.eu/content/d31-framework-assessment-methods-and-tools>
- [7] D3.2 Catalogue of evaluated methodologies and tools available (2011), <http://www.valuesec.eu/content/d32-catalogue-evaluated-methodologies-and-tools-available>
- [8] D3.3 Evaluation of methods and tools, and the required improvements (2012), <http://www.valuesec.eu/content/d33-evaluation-methods-and-tools-and-required-improvements>
- [9] D4.1 Part 1 Usability assessment criteria and usability analysis (2012), <http://www.valuesec.eu/content/d41-part-1-usability-assessment-criteria-and-usability-analysis>
- [10] Zuniga, E.B., Blobner, C.: ValueSec – Mastering the Value Function of Security Measures. In: Ender, J., Fiege, J. (eds.) 6th Future Security: Security Research Conference, Future Security, Berlin, September 5-7. Conference Proceedings, pp. 277–281 (2011)
- [11] BJORHEIM ABRAHAMSEN, E., AVEN, T., PETTERSEN, K., ROSQVIST, T.: A framework for selection of strategy for management of security measures. In: PSAM 2011 & Esrel 2012 Int'l Conference Proceedings. Scandic Marina Congress Centre, Helsinki, Finland, June 25-29, pp. 18-Tu2-4. USB memory stick (2012)
- [12] RÄIKKÖNEN, M., ROSQVIST, T., POUSSA, L., JÄHI, M.: A Framework for Integrating Economic Evaluation and Risk Assessment to Support Policymakers' Security-related Decisions. In: PSAM 2011 & Esrel 2012 Int'l Conference Proceedings. Scandic Marina Congress Centre, Helsinki, Finland, June 25-29, pp. 18-Tu3-2. USB memory stick (2012)
- [13] ADAR, E., BLOBNER, C., HUTTER, R., PETTERSEN, K.: An extended Cost-Benefit Analysis for evaluating Decisions on Security Measures of Public Decision Makers. Forthcoming CRITIS 2012, 7th International Conference on Critical Information Infrastructures Security, Lillehammer, Norway, September 17-19 (2012)
- [14] BIAŁAS, A.: Computer support in business continuity and information security management. In: KAPCZYŃSKI, A., TKACZ, E., ROSTANSKI, M. (eds.) Internet - Technical Developments and Applications 2. AISC, vol. 118, pp. 155–169. Springer, Heidelberg (2012)
- [15] BAGAŃSKI, J., ROSTAŃSKI, M.: The modeling of Business Impact Analysis for the loss of integrity, confidentiality and availability in business processes and data. *Theoretical and Applied Informatics* 23(1), 73–82 (2011) ISSN 1896-5334
- [16] BAGAŃSKI, J.: Software support of the risk reduction assessment in the valueSec project flood use case. In: ZAMOJSKI, W., MAZURKIEWICZ, J., SUGIER, J., WALKOWIAK, T., KACPRZYK, J. (eds.) *New Results in Dependability & Comput. Syst.* AISC, vol. 224, pp. 11–24. Springer, Heidelberg (2013)

Reduction of Computational Cost in Mutation Testing by Sampling Mutants

Ilona Bluemke and Karol Kulesza

Institute of Computer Science,
Warsaw University of Technology, Nowowiejska 15/19,
00-665 Warsaw, Poland
I.Bluemke@ii.pw.edu.pl

Abstract. The objective of this chapter is to explore the reduction of computational costs of mutation testing by randomly sampling mutants. Several experiments were conducted in the Eclipse environment using *MuClipse* and *CodePro* plugins and especially designed and implemented tools: *Mutants Remover* and *Console Output Analyser*. Six types of mutant' subsets were generated and examined. Mutation score and the source code coverage were used to evaluate the effectiveness of mutation testing with subsets of mutants. The ability to detect errors introduced "on purpose" in the source code was also examined.

1 Introduction

Mutation testing is a fault based software testing technique that was introduced more than forty years ago. The general idea is that the faults used in mutation testing represent the mistakes made by a programmer so they are deliberately introduced into the program to create a set of faulty programs called *mutants*. Each mutant program is obtained by applying a mutant operator to a location in the original program. To assess the quality of a given set of tests these mutants are executed against the set of input data to see, if the inserted faults can be detected. A very good survey of mutation techniques was written in 1996 by Jia and Harman [1], they also created a repository [2] containing many interesting papers on mutation testing. Recently Bashir and Nadeem published a survey on object mutation [3].

Mutation testing is effective at measuring the adequacy of a test suite, but it can be computationally expensive to apply all the test cases to each mutant. Previous research has investigated the effect of reducing the number of mutants by selecting certain operators, sampling mutants at random, or combining them to form new higher-order mutants.

The objective of this chapter is to examine what is the impact of randomly sampling mutants for Java programs, on the mutation score, the code coverage and the ability to detect real errors. The main ideas of mutation testing and reducing the number of mutants are briefly described in section 2 while related work is presented in section 3. The results of experiments are presented in section 4 and some conclusions are given in section 5.

2 Mutation Testing

The mutation testing is a fault based software testing technique that was introduced in 1971 by Richard Lipton (according to [4]). Surveys on mutation techniques were written e.g. by Jia and Harman [1], Bashir and Nadeem [3]. Many papers on mutation testing can be found in a repository [2].

The general idea of mutation testing is that the faults represent mistakes made by a programmer, so they are deliberately introduced into the program to create a set of faulty programs called *mutants*. Each mutant program is obtained by applying a mutant operator to a location in the original program. Typical mutation operators include replacing one operator e. g. '+' by another e.g. '-' or replacing one variable by another. To assess the quality of a given set of tests the mutants are executed on a set of input data to see, if the inserted faults can be detected. If the test is able to detect the change (i.e. one of the tests fails), then the mutant is said to be *killed*. The input data for test should cause different program states for the mutant and the original program.

A variety of mutation operators were explored by researchers. Some examples of mutation operators for imperative languages: statement deletion, replacement of each Boolean sub expression with *true* and *false*, replacement of each arithmetic operation with another one, e.g.: "*" with "/", replacement of each Boolean relation with another one, e.g.: > with >=, == .

These mutation operators are also called traditional mutation operators. There are also mutation operators for object-oriented languages, for concurrent constructions, complex objects like containers etc., they are called class-level mutation operators. In [3] a survey on the existing object oriented mutation techniques is presented. These techniques are critically reviewed on the basis of evaluation criteria designed by the authors by considering important aspects of mutation testing. These aspects can have their influence on mutation testing process. Another contribution of this work is a survey of available mutation testing tools.

One of the greatest challenges to the validity of mutation testing is the number of mutants that are semantically equivalent to the original program. *Equivalent mutants* produce the same output as the original program for every possible input. For seven large Java programs, 45% of the mutants not detected by the test suite were shown to be equivalent [5]. Equivalent mutants occur when the mutation can never be exercised, its effect is later cancelled out or it is corroded away by other operations in the program [6]. Determining which mutants are equivalent is a tedious activity, usually not implemented in tools. The impact of equivalent mutants is studied in [7]. Techniques have been devised to identify equivalent mutants using program slicing [8], compiler optimization [9], constraint solving [10] and, more recently, impact assessment [7]. Equivalent mutants are still however difficult to remove completely.

Mutation score is a kind of quantitative test quality measurement that examines a test set's effectiveness. It is defined as a ratio of the number of killed mutants to the total number of non-equivalent mutants. The total number of nonequivalent

mutants results from the difference between the total number of mutants and the number of equivalent mutants which cannot be killed.

Mutation testing of software would be difficult without a reliable, fast and automated tool that generates mutants, runs them against a test suit and reports the results. Among several Java mutation tools there are e.g. Muclipse [11], Judy [12], JavaLanche [7].

The number of generated mutants depends on the number of mutation operators available in a mutation tool e.g. in Muclipse there are 43 mutation operators while in Jumble [13], also Java mutation tool, only 7. It is not feasible to use every possible mutant of the program under test, even after all the equivalent mutants have been removed. It is therefore necessary to select a subset of mutants that allow the test suite to be evaluated within a reasonable period of time. Some research has been conducted to reduce the number of mutants by selecting certain operators, sampling mutants at random, or combining them to form new higher-order mutants. *Mutant sampling* was proposed by Acree [14] and Budda [15] in 1980. Firstly all mutants are generated, then randomly a subset of mutants is chosen and the remaining ones are ignored.

3 Related Work

The problem of reducing the cost of mutation testing was studied in several papers. In [16] Mathur and Wong proposed two techniques limiting the number of mutants:

1. randomly selected $x\%$ mutants (starting from 10%, then 15%, 20%, 25%, 30%, 35% and 40%),
2. the constrained mutation - only a few specific types of mutants are used (*abs* - absolute value insertion and *ror* - relational operator replacement) and others are ignored.

In experiments they shown that both approaches lead to test sets that distinguish a significant number of all mutants and provide high coverage. Four Fortran programs and Mothra [17] tool were used. For full set of mutants and for each subsets, test cases killing all nonequivalent mutants were generated and cost reduction were calculated. Cost reduction can be calculated as the proportion of mutants in the sample to the total number of nonequivalent mutants. Mathur and Wong shown that using only mutants *abs* and *ror* reduces the number of test cases 40%-58%. Similar results were obtained for subsets containing 10% to 40% of total number of mutants. The number of test cases was reduced 24%-63%. They compared also the cost of creating test cases. For randomly selected $x\%$ mutants this cost was decreased at least 60% and for constrained mutants at least 80%. This gain was accompanied by a small loss in the ability to distinguish nonequivalent mutants and code coverage.

Slightly different approach to mutants' sampling was proposed in [18]. Scholive, Beroulle and Robac proposed to choose a subset of mutants generated for

each mutation operator. For four programs used in the experiment a subset containing 10% of mutants was created. The testing results for the complete set of mutants and the subset were better (more mutants were killed) than testing with randomly chosen mutants.

Offutt, Rothermel and Zapf in [19] were examining constrained mutation (some mutation operators were ignored). They were using Mothra [17] tool with 22 mutation operators divided into three categories:

1. ES (Expression/Statement) – change of an operator or modification of the whole expression.
2. RS (Replacement/Statement) – change of operands or modification of the whole expression.
3. RE (Replacement/Expression) – change of operators or operands.

Ten Fortran programs (with 10 to 48 lines of code) were used in experiment. Using the above listed groups of mutation operators mutants were generated, also equivalent mutants were in these sets. Godzilla [20] tool was used to create test cases killing all nonequivalent mutants. These test cases were run on mutants generated by all 22 mutation operators. The number of killed mutants were compared. The experiment showed high values of killed mutants in each category: ES – 99.54%, RS – 97.31%, RE – 99.97%. Analyzing the results of this experiments authors proposed the fourth category – E – of mutation operators, ignoring operators modifying operands or expressions. However this category contained only 5 operators, for ten tested programs the minimal factor of killed mutants was 98.67%, the average was 99.51%. Selective mutation decreased the number of mutants by 44% in average.

Using the results of the above described experiments and performing others experiments Mresa and Bottaci proposed in [21] set of efficient operators – *eff*. This set contains operators: statement analysis – *san*, arithmetic operator replacement – *aor*, statement deletion – *sdl*, relational operator replacement – *ror*, unary operator insertion – *uoi*. To the *eff* set the absolute value insertion operator – *abs* was added and the new set was named *efa* (efficient operators + abs). Mresa and Bottaci also used proposed in [19], set E – omitting operators modifying operand or whole expressions. They named this set as *exp* (expression). The average values of mutants killing factors were really high: *eff* – 99.2%, *efa* – 99.6% and for *exp* – 99.7%.

Another approach to the mutant reduction problem was proposed by Patrick, Oriol and Clark in [22]. They propose to use static analysis to reduce mutants. Symbolic representations are generated for the output along the paths through each mutant and these are compared with the original program. By calculating the range of their output expressions the effect of each mutation on the program output is determined. Mutants with little effect on the output are harder to kill. They confirm this using random testing and an established test suite. In their experiments, the average mutant in the top quarter is less than half as likely to be killed by a random test case than the average mutant in the remaining three quarters. This technique provides an independent selection of semantically small mutations and forgoes the expense of evaluating mutants against a test suite.

4 Experiment

The goal of the experiment was to explore random sampling mutations. Our experiments were conducted in the Eclipse environment. *MuClipse* [11] and *CodePro* [23] plugins were used for the mutation testing. Two special tools: *Mutants Remover* [24] and *Console Output Analyzer* [24] were designed and implemented especially for this experiment. Eight Java classes (listed in Table 1), open source or written by students of Institute of Computer Science, were tested. For these classes 53 to 556 mutants were generated. The classes were tested on laptop Benq-JoybookR55 with processor 1.60 Ghz and 2 GB RAM under Windows 7. Some classes, for which the time of tests was greater than 10 min, were tested on computer with AMD Athlon 4x processor, 3 Ghz and 4 GB RAM.

Table 1 Tested classes

<i>class</i>	<i>Project</i>	<i>source</i>	<i>Number of methods</i>	<i>Lines of code</i>	<i>Number of mutants/equivalent mutants</i>
Recipe	CoffeeMaker	[25]	14	84	138/15
CoffeeMaker	CoffeeMaker	[25]	8	102	285/17
Money	CodePro JUnit Demo	[26]	14	59	53/4
MoneyBag	CodePro JUnit Demo	[26]	17	114	54/6
Element	MapMaker	[27]	10	80	380/20
Board	NetworkShip-Battle	[28]	12	123	270/3
Wall	jet-tetris	[29]	7	79	290/19
Stack	javasol	[30]	26	176	556/30

4.1 Experiment Method

For each class, being the subject of our experiment, firstly all mutants were generated. Secondly the test cases killing these mutants were generated using *JUnit*, part of *CodePro* plugin. *Console Output Analyzer* [24] was identifying test cases not killing mutants. Time consuming was the identification of equivalent mutants, based on the analysis of source code of the original program and its mutants. It was also always necessary to construct test cases “by hand” for several nonequivalent mutants to obtain an adequate test set. The number of test cases generated automatically by *CodePro* was only 28.78% so quite a lot of time was spend on constructing test cases manually. The number of mutants killed by automatically generated tests was 47.15%. Based on the results of Mresa’ and Bottaci’ research [21] *effective test sequence* were built. Informally, each test in an effective sequence is non-redundant with respect to the tests that precede it.

The initial set of all generated mutants was reduced by sampling and selective mutations. In this chapter the experimental results only of sampling mutants are presented. In sampling mutants randomly only subsets of all mutants containing

60%, 50%, 40%, 30%, 20% and 10% of all are selected while remaining mutants are ignored. *Mutants Remover* tool [24] was used in building appropriate sets of mutants. The choice of the number of random samples should be made from a significantly “big” set. Due to time limitations and the effort needed to construct test cases and identify equivalent mutants, for each class being the subject of this experiment, only 18 sets of mutants were constructed which is not sufficient to obtain statistically correct results.

In next step test cases “killing” all mutants in the set were produced. Firstly the *CodePro* generator was generating test cases and *Console Output Analyzer* [24] was identifying test cases not killing mutants. For the not “killed” mutants the test cases prepared for the whole set of mutants were used.

The sets of mutants and theirs test cases were next evaluated using following two criteria:

- I. “killed” mutants factor. It was assumed that test cases killing 95% of all mutants are adequate.
- II. code coverage. *CodePro* plugin was measuring the code coverage for test cases prepared for all mutants and each subset. We assumed that 2% decrease of the code coverage is acceptable.

In the next phase of the experiment we were checking the ability to detect errors introduced “on purpose” by students of the Institute of Computer Science Warsaw University of Technology. For each class, being the subject of our experiment, many versions containing errors were stored. For each version the test cases were run and the number of detected errors was calculated.

4.2 Experiments Results

In Table 2 mutants killed factors are shown for randomly sampled mutants. In columns the percentage of sampled mutants is given. In bold font the values exceeding 95% are presented, satisfying criterion I (section 4.1). The average values exceeding 95% for all examined classes were obtained only for subsets SAMP_60 and SAMP_50 containing accordingly 60% and 50% of mutants. Among all 48 values of killed factor the level 95% was achieved in 21 cases.

Table 2 Percentage of killed mutant in SAMP subsets

Class/subset	60%	50%	40%	30%	20%	10%
Recipe	93.77	92.68	90.51	88.35	83.20	62.87
CoffeeMaker	98.88	98.13	98.51	97.89	95.77	91.54
Money	95.92	90.48	87.76	84.35	82.31	60.54
MoneyBag	96.53	95.14	93.75	93.06	84.03	63.16
Element	98.24	96.11	96.48	93.33	92.22	83.52
Board	98.38	97.63	97.63	94.268	94.13	82.02
Wall	99.75	99.26	99.02	96.68	96.92	91.64
Stack	98.42	97.59	94.36	93.28	86.31	70.41

In Table 3 the code coverage for tested classes is shown. For majority of classes the code coverage is greater than 90%. For some classes not all methods in class are covered (e.g. MoneyBag, Stack) because *MuClipse* didn't generated mutants for these methods.

Table 3 Code coverage for test cases for all mutants

class	Coverage for all mutants	Number of methods	Number of covered methods
Recipe	95.90%	14	14
CofeeMaker	98.20%	8	7
Money	84%	14	10
MoneyBag	74.20%	17	13
Element	99%	10	10
Board	99%	12	12
Wall	100%	7	7
Stack	94.50%	26	23

In Table 4 the decrease in the code coverage for SAMP subset, related to code coverage for test cases prepared for all mutants and presented in Table 3, is shown. We assumed that the decrease 2% is acceptable and such values are given in bold fonts in Table 4. The criterion II (section 4.1) was fulfilled for 14 cases out of 48. The average values of the code coverage decrease were also calculated. Only for subsets SAMP_60 the average (for all classes) decrease in code coverage was less than 2%, i.e. 1.45%. So only this subset can be adequate according to criterion II (section 4.1).

Table 4 Decrease in code coverage for SAMP subsets

Class/subset	60%	50%	40%	30%	20%	10%
Recipe	3.37%	4.40%	6.10%	8.47%	13.27%	33.33%
CofeeMaker	0%	0.40%	0%	0%	2.23%	7.23%
Money	2.07%	7.17%	8.53%	10.87%	12.93%	31.20%
MoneyBag	0.43%	0.43%	2.80%	1.63%	6.73%	19.97%
Element	0.70%	1.17%	0.93%	2.90%	2.57%	4.53%
Board	2.97%	3.57%	3.67%	10.53%	11.97%	27.33%
Wall	0%	0.27%	0%	0.83%	0.53%	8.37%
Stack	2.07%	2.40%	4.10%	3.37%	4.87%	11.20%

Each of mutants has to be executed at least once for a test case. If the number of test cases decreases the time to test mutants also decreases. On the other hand greater number of test cases enables better testing of the source code. In Table 5 the number of test cases for all mutants and for subsets are presented. For each

subset the average value for three subsets is given. In majority of subsets the reduction of the cardinality of test cases is not very convincing. The values written in bold fonts denote that for these subset both evaluation criteria (section 4.1) are met.

Table 5 Number of test cases for mutants sets

Class/subset	100%	60%	50%	40%	30%	20%	10%
Recipe	29	21.67	20.33	16.67	13.67	11	8
CofeeMaker	25	21	19.67	19.33	16.67	14.33	10.33
Money	17	11.67	10	9.67	8	7.33	5
MoneyBag	13	10.67	9.67	10	8.67	5.67	4
Element	35	30.33	28.33	28.67	26.67	23.33	17.33
Board	28	22.67	22.33	21	18.33	12.67	9.67
Wall	19	17.33	16	15.33	12	12.33	9.33
Stack	83	74.33	69.67	63.67	59.67	52	35.67

Lower number of mutants and lower number of test cases decrease also the total number of program executions in testing process thus decreasing the time to test. The number of program runs were measured by *Console Output Analyzer*. Sampling 60% of mutants needed only 47.77% of runs for all mutants. For 50%, 40%, 30%, 20% and 10% the reduction were accordingly: 59.12%, 69.16%, 79.29%, 87.80% and 95.18%.

In Table 6 the second column contains the number of errors introduced on purpose by students, in each class being the subject of our experiment. In the third column the percentage of errors detected by test cases constructed for all mutants are given. Mutation testing was very successful in detecting these errors – 96.15% was the average value calculated for all tested classes. Sampling 60% of mutants detected in average 91.74% while the average values for 50%, 40%, 30%, 20% and 10% are accordingly: 88.73%, 87.78%, 85.74%, 84.46%, 72.98%.

Table 6 Percentage of detected errors

Class/subset	Errors introduced	100%	60%	50%	40%	30%	20%	10%
Recipe	24	87.50	70.83	72.22	70.83	63.89	66.67	48.61
CofeeMaker	29	100	100	98.85	100	100	98.85	93.10
Money	10	100	96.67	76.67	76.67	83.33	83.33	63.33
MoneyBag	14	85.71	85.71	85.71	83.33	83.33	76.19	78.57
Element	20	100	95	93.33	95	81.67	85	66.67
Board	26	100	91.03	89.74	88.46	85.90	84.62	79.49
Wall	24	100	100	100	98.61	95.83	91.67	84.72
Stack	25	96	91.74	88.73	87.78	85.74	84.46	72.98

5 Conclusions

Experimental research has shown mutation testing to be very effective in detecting faults [6,31,32,33], unfortunately it is computationally expensive. The goal of our research was to examine randomly sampling mutants in object programs. The previous experiments were made with structural languages, mainly Fortran. Our experiment, described in section 4, shows that randomly sampling 60% or 50% of mutants in Java programs can significantly reduce the cost of testing with acceptable mutation score and code coverage. Also these subsets of mutants were effective in detecting errors introduced by users. The experiments reported in this chapter were very time consuming so only 8 Java classes were tested by sampling mutations. The number of programs used in other experiments on mutation' subset were similar. It is difficult to know if 8 classes is sufficiently large sample from which to generalize and so similar studies on larger sets of classes will be useful. Due to the effort needed in performing the experiment we were not able to use statistically significant number of mutants for random selection. This can be seen in the results of some experiments e.g. in Table 6 for class Recipe: a 20% sample was able to find more faults than a 30% sample.

All the results of this study have been obtained using the set of mutation operators used by *MuClipse*. Clearly, these results cannot be applied directly to mutation systems that use different operators. Efficiency relationships will, nonetheless, be present between any set of operators.

Acknowledgments. We are very grateful to the reviewers for many valuable remarks.

References

- [1] Jia, Y., Harman, M.: An Analysis and Survey of the Development of Mutation Testing. Technical report TR-09-06, Crest Centre, Kong's College London (1996), <http://www.dcs.kcl.ac.uk/pg/jiayue/repository/TR-09-06.pdf> (accessed 2010)
- [2] Mutation repository, <http://www.dcs.kcl.ac.uk/pg/jiayue/repository> (accessed 2012)
- [3] Bashir, B.M., Nadeem, A.: Object Oriented Mutation Testing: A Survey. IEEE (2012), 978-1-4673-4451-7/12
- [4] Mathur, A.P.: Mutation Testing. In: Marciniak, J.J. (ed.) Encyclopedia of Software Engineering, pp. 707–713 (1994)
- [5] Schuler, D., Zeller, A.: (Un-)covering equivalent mutants. In: Proc. ICST, pp. 45–54 (2010)
- [6] Voas, J.M., Miller, K.W.: Software testability: the new verification. IEEE Softw. 12(3), 17–28 (1995)
- [7] Grun, B., Schuler, D., Zeller, A.: The impact of equivalent mutants. In: Proceedings of the 4th International Workshop on Mutation Testing (2009)

- [8] Hierons, R., Harman, M., Danicic, S.: Using program slicing to assist in the detection of equivalent mutants. *Softw. Test. Verif. Rel.* 9(4), 233–262 (1999)
- [9] Offutt, A.J., Craft, W.M.: Using compiler optimization techniques to detect equivalent mutants. *Softw. Test. Verif. Rel.* 4(3), 131–154 (1994)
- [10] Offutt, A.J., Pan, J.: Automatically detecting equivalent mutants and infeasible paths. *Softw. Test. Verif. Rel.* 7(3), 165–192 (1997)
- [11] MuClipse (2012), <http://muclipse.sourceforge.net/index.php> (accessed 2012)
- [12] Madeyski, L., Radyk, R.: Judy - A Mutation Testing Tool for Java. *IET Software* 4(1), 32–42 (2010), doi:10.1049/iet-sen.2008.0038
- [13] Jumble, <http://jumble.sourceforge.net/index.ht> (accessed 2010)
- [14] Acree, A.T.: On mutation. Ph.D. thesis, Georgia Institute of Technology, Atlanta, Georgia (1980)
- [15] Budd, T.A.: Mutation analysis of program test data. Ph.D. thesis, Yale University, New Haven, Connecticut (1980)
- [16] Mathur, A.P., Wong, W.E.: Reducing the cost of mutation testing: an empirical study. *J. Syst. Softw.* 31(3), 185–196 (1995)
- [17] DeMillo, R.A., Guindi, D.S., King, K.N., McCracken, W.M., Offutt, A.J.: An extended overview of the Mothra software testing environment. In: *Proc. of the Second Workshop on Software Testing, Verification, and Analysis*, pp. 142–151 (July 1988)
- [18] Scholive, M., Beroulle, V., Robach, C.: Mutation Sampling Technique for the Generation of Structural Test Data. In: *Proc. of the Conf. on Design, Automation and Test in Europe*, vol. 2, pp. 1022–1023 (March 2005)
- [19] Offutt, J., Rothermel, G., Zapf, C.: An experimental determination of sufficient mutation operators. *ACM Trans. on Soft. Eng. and Methodology* 5(2), 99–118 (1996)
- [20] DeMillo, R.A., Offutt, A.J.: Constraint-based automatic test data generation. *IEEE Trans. on Soft. Eng.* 17(9), 900–910 (1991)
- [21] Mresa, E.S., Bottaci, L.: Efficiency of mutation operators and selective mutation strategies: An empirical study. *Soft. Testing, Ver. and Rel.* 9(4), 205–232 (1999) ISSN: 09600833
- [22] Patrick, M., Oriol, M., Clark, J.A.: MESSI: Mutant Evaluation by Static Semantic Interpretation. In: *2012 IEEE Fifth Int. Conf. on Software Testing, Verification and Validation*, pp. 711–719 (2012), doi:10.1109/ICST.2012.161
- [23] CodePro JUnit Demo (2012), <https://developers.google.com/java-dev-tools/codepro/doc/features/junit/CodeProJUnitDemo.zip> (accessed 2012)
- [24] Kulesza, K.: Mutation testing computational cost reduction using mutants sampling and selective mutation. M.Sc. thesis, Institute of Computer Science, Warsaw University of Technology (September 2012)
- [25] CoffeeMaker, http://agile.csc.ncsu.edu/SEMaterials/tutorials/coffee_maker (accessed April 2012)
- [26] CodePro JUnit Demo, <https://developers.google.com/java-dev-tools/codepro/doc/features/junit/CodeProJUnitDemo.zip> (accessed 2012)

- [27] Godlewski, Ł.: MapMaker. Institute of Computer Science, Warsaw University of Technology (2008)
- [28] Suchowski, J.: Network game – NetworkShipsBattle. Institute of Computer Science, Warsaw University of Technology (2010)
- [29] jet-tetris,
<http://sourceforge.net/projects/jet-tetris> (accessed 2012)
- [30] javasol,
<http://sourceforge.net/projects/javasol> (accessed May, 2012)
- [31] Bluemke, I., Kulesza, K.: A Comparison of Dataflow and Mutation Testing of Java Methods. In: Zamojski, W., Kacprzyk, J., Mazurkiewicz, J., Sugier, J., Walkowiak, T. (eds.) Dependable Computer Systems. AISC, vol. 97, pp. 17–30. Springer, Heidelberg (2011)
- [32] Frankl, P.G., Weiss, S.N., Hu, C.: All-uses versus mutation testing: an experimental comparison of effectiveness. *J. Syst. Softw.* 38(3), 235–253 (1997)
- [33] Andrews, J.H., Briand, L.C., Labiche, Y.: Is mutation an appropriate tool for testing experiments? In: *Proc. ICSE*, pp. 402–411 (2005)

Use of Neural Network Algorithms in Prediction of XLPE HV Insulation Properties under Thermal Aging

Boukezzi Larbi¹ and Boubakeur Ahmed²

¹Materials Science and Informatics Laboratory, MSIL,
Djelfa University, Algeria,
University Ziane Achour of Djelfa,
BP 3117 Route of Moudjbara,
Djelfa 17000, Algeria

lboukezzi@mail.univ-djelfa.dz

²L.R.E./High voltage laboratory of ENP,
ENP of Algiers, Algeria,
National Polytechnic School of Algiers,
B.P182, EL-Harrach, Algiers, Algeria
ahmed.boubakeur@enp.edu.dz

Abstract. Some Artificial neural network algorithms have been used to predict properties of high voltage electrical insulation under thermal aging in term to reduce the aging experiment time. In this work we present a short comparison of the obtained results in the case of Cross-linked Polyethylene (XLPE). The theoretical and the experimental results are concordant. As a neural network application, we propose a new method based on Radial Basis Function Gaussian network (RBFN) trained by two algorithms: Random Optimization Method (ROM) and Back-propagation (BP).

1 Introduction

Electric material manufacturers constantly search for new and improved material to satisfy electric power customers' demands for continuous supply of quality electric power. In particular, dielectric materials have earned a privileged place. In practice, the insulations used in high voltage (HV) technology, particularly liquid and solid insulation, lose their mechanical, electrical and physical properties under the temperature constraint [1-4]. An elevated temperature for a long time duration leads to a degradation of the insulating material, which reduces the equipment life duration.

The great importance of XLPE as insulation encourages researchers in laboratories through the world to investigate a lot of experimental techniques in order to get more informations and characterize well the degradation mechanisms of the material under service conditions. These investigations are costly and time

consuming. They need sometimes few years to get sufficient database to solve economical problems of energy and making maintenance in a simplest way.

Studies had been carried out by us [5] at the HV Laboratory of the ENP of Algiers on XLPE insulation cables and focused on the thermal aging. The test-probes experiments on XLPE were effectuated under different temperatures included between 80°C and 140°C with a maximum aging duration time of 5000 hours. These tests consisted of aging the material and measuring some dielectric and mechanical properties after regular time intervals [5,6]. The variation of these properties versus the time, for different constraint values, allows us to establish the life duration and the material strength characteristic.

Recently, several intelligent systems have been developed which help scientist and engineer to use in efficient way the database and get with more precision future state of the insulation system. The use of Artificial Neural Network (ANN) presents a powerful tool to predict and diagnosis high voltage insulation materials. In recent studies, this tool was applied in the pattern recognition of partial discharge pulse [7,8], classification and diagnosis of transformer oil [9], and modelling of pollution flashover on high voltage insulators [10].

This work is focussed on the use of some algorithms of artificial neural network to predict the XLPE behaviour under thermal stress in order to reduce the aging experiments time and to compare their performances of prediction. To come up at this objective we have used supervised neural networks based on Radial Basis Function Gaussian (RBF) trained with two algorithms: Back-propagation (BP) and Random Optimization Method (ROM). These algorithms ensure a good prediction with a relative error of 5%.

2 Prediction Method

To predict mechanical properties of XLPE insulation cables under thermal aging, we have used neural networks based on radial basis function (RBF) (figure 1): the used neural network contains an input layer with n units (aging data), a hidden layer with m units and an output layer with a single node (predicted value of property).

The use of Gaussian function allows its local characteristics to facilitate the training and improve generalisation. The main idea of the RBF networks [11] is that any function $f(x)$ can be approximated by an interpolation composed by the sum of p core functions:

$$f(x) = \sum_{i=1}^p w_j \phi(|x - \xi_i|) \quad (1)$$

Where ξ_i are the nodes of interpolation for $i=1, n$ called centers and w_j are the synaptic weights that interconnect neurons to the output. ϕ is the core function, it ensures the continuity in the nodes.

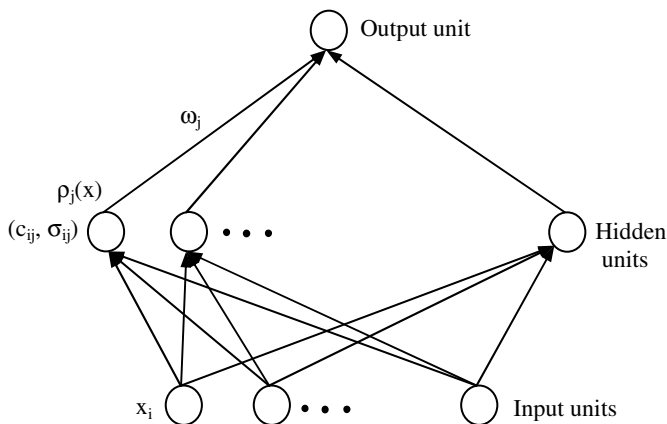


Fig. 1 A feed-forward network with a single hidden layer and a single output unit

Each node in the hidden layer has a radial symmetric response around a node parameter vector called center. The output layer is a linear combiner with connection weights [12]. Giving a set of input and output data $(x_i, y_i)_{i=1,n}$, the state of the hidden unit (j) will be denoted by:

$$\rho_j(x_i) = \exp\left(-\frac{1}{2} \sum_{i=1}^n \frac{(x_i - c_{ij})^2}{\sigma_{ij}^2}\right) \tag{2}$$

Where c_{ij} $i=1,n$ and $j=1,m$ are the RBFG centers, σ_{ij} define the width of Gaussians.

The chosen network in our investigation is trained by two different methods: Random Optimization Method (ROM) and Back-Propagation algorithm (BP).

2.1 RBFG Trained by ROM

The RBFG centers are vectors of n dimensions; they can be selected from the trained data by some mechanisms cited in [11]. In the training by ROM proposed in this work, the used technique consists of an arrangement of centers in a regular trellis in order to cover uniformly the data input space.

2.2 RBFG Trained by Back-Propagation

Training radial basis network with back-propagation algorithm uses the gradient descendant method. It requires a choice of learning rate for adjustment of the weights w_j , gaussian centers c_{jk} and variances σ_{jk} . These adjustments are one of the causes of local minima because of their dependence on the error gradient. The adaptation of w_j , c_{jk} and σ_{jk} is given respectively by:

$$\left\{ \begin{array}{l} \Delta w_j = -\eta_w \frac{\partial J}{\partial w_j} \\ \Delta c_{jk} = -\eta_c \frac{\partial J}{\partial c_{jk}} \\ \Delta \sigma_{jk} = -\eta_\sigma \frac{\partial J}{\partial \sigma_{jk}} \end{array} \right. \quad (5)$$

And:

$$J = \frac{(y - y^*)^2}{2}$$

Where:

y is the target and y^* is the net output

η_w , η_c and η_σ are the learning rates to adjust respectively weights, centers and variances.

The prediction process is based on two main phases: the training and the prediction.

3 Training Phase

The used technique to train the ROM network is the FFN Pattern (data adaptive learning). In this case, the training is done on a set of samples having the form (y_i, y_{i+1}) , where y_i is a property value corresponding to aging time t_i , and y_{i+1} is the predicted value.

4 Prediction Phase

In order to predict a future value y_{n+1} of a set of measured data y_i , $i=1, n$, the algorithm will be trained on a set of samples having form $(y_i, y_{i+1})_{i=1, n-1}$. After this training, the weights of the net are updated so that when the network receives the value y_i , its response will be y_{i+1} . Then, a new set of data is obtained with $n+1$ values. The training is repeated from the beginning in order to predict the value y_{i+2} , the new set of data will contain $n+2$ values, and this procedure is repeated until the prediction of the property for the entire required aging interval. In order to improve the prediction quality, after each future value prediction ($n+1$), the first value (1) is omitted from the set of data. In this way, the network is trained by the same number of input data.

5 Results and Discussion

In this section we have used neural networks trained by ROM and BP as described in sections 2.1 and 2.2. In a first step we have trained the NN with ROM using FFN pattern for two learning times 2000h and 2500h to predict tensile strength under thermal aging at 80°C and 100°C. Results show the influence of the learning time on the network learning quality. From the obtained results (figure 2), the NN predicts well with a learning time of 2000h than 2500h. This result indicates that there is an optimum learning time to get a better learning quality, and leads to ascertain also the existence of an optimum time step and an optimum learning time giving the best result but not together. The maximum error of 2000h learning time is about 5% in the case of 80°C and 6.3% in the case of 100°C and reaches 18.9% and 28.9% for 80°C and 100°C respectively. It is very interesting to note that the error function has a local minimum as a function of learning time, which in turn depends on the time step.

For purpose of illustration and comparison, we present results of prediction of each studied material’s property using ROM trained by FFN pattern and back-propagation in the same figure. All the results are obtained with a relative error of 5%. By fixing the learning step to 500 hours, we trained the net for a learning time less than the maximum aging time for each temperature, and then the neural network predicts each property at the whole experimental interval.

This will allow us to validate our prediction quality at the cited interval. For aging temperatures 80°C and 100°C we used a learning time of 2000h. For the other temperatures we have used a learning time of 1500h. Figures 3(a,b,c,d) and 4(a,b,c,d) show the measured and predicted values of elongation at rupture and tensile strength of XLPE regarding the aging time.

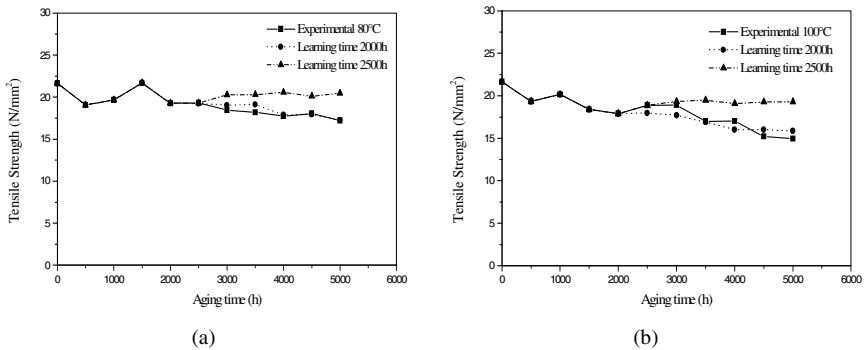


Fig. 2 Measured and predicted values of tensile strength according to aging time using two learning times

- (a) aging temperature 80°C,
- (b) aging temperature 100°C.

The obtained results pointed out that both of ROM and BP are good prediction neural networks and present acceptable errors. However, the ROM presents the best quality of prediction because of its independence of the used mathematical model and for its ability to adjust the weights without determining any gradient. Moreover, the use of BP network presents a lot of difficulties in the training (adjustment of many parameters in the same time) which is very difficult and takes a long time in the generalization step.

The reduced precision of the prediction especially in the case of BP algorithm is due to several factors. The smaller sample of data is in the first position. By analyzing this obstacle, we can say that the application of neural network to relatively short data records to an illconditioned problem. In this case, as the cost function is full of deep local minima, the optimal search would likely end up trapped in one of the local minima. The situation gets worse when either (i): make the neural network more nonlinear (by increasing the number of neurons, hence the number of parameters), or (ii): shorten the data size. A simple way to deal with the local minima problem is to use a neural network where their parameters are randomly initialized before training.

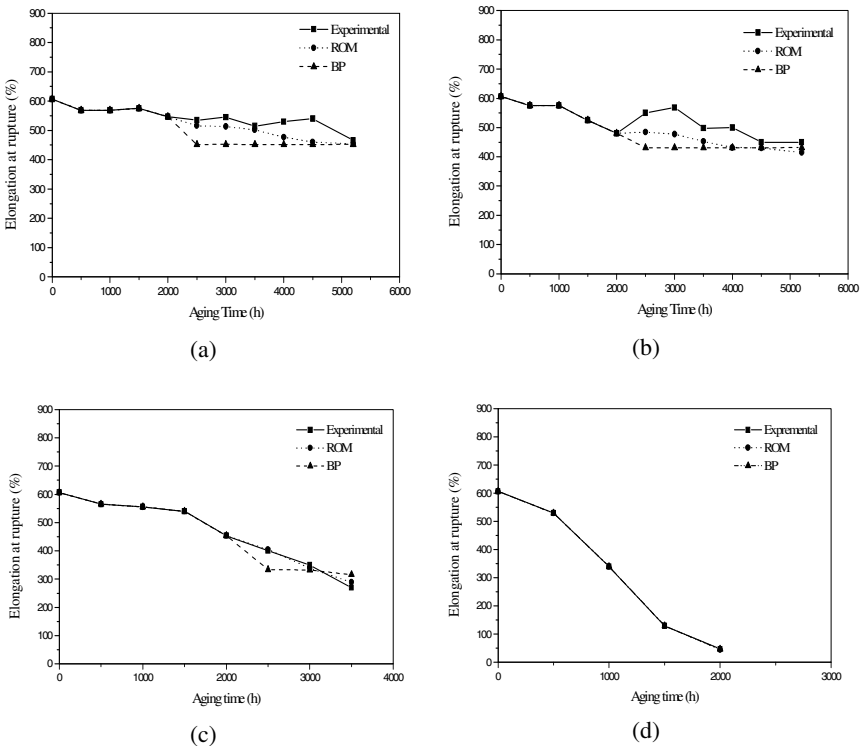


Fig. 3 Measured and predicted values of elongation at rupture according to aging time: (a): 80°C, (b): 100°C, (c): 120°C and (d): 140°C

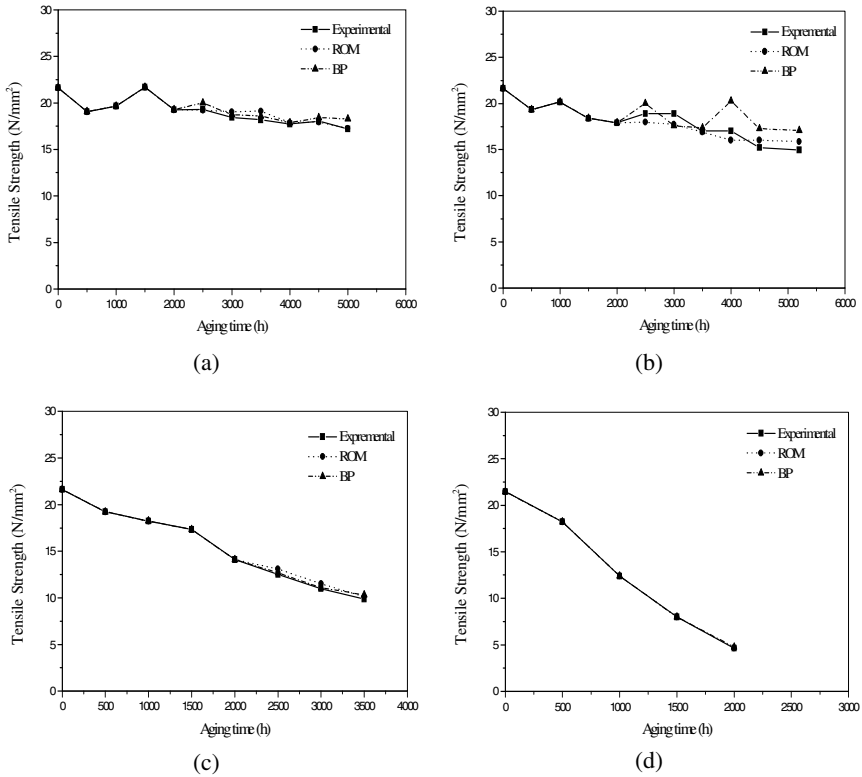


Fig. 4 Measured and predicted values of tensile strength according to aging time: (a): 80°C, (b): 100°C, (c): 120°C and (d): 140°C

We farther analyzed the erroneously predicted values for the two used methods in term of error and we present the error of prediction in Tables 1 and 2. It is clearly seen that the prediction performances of ROM are better than of BP except some values of prediction as we evoked previously. In the case of elongation at rupture, the maximum relative error reaches at 80°C 14.81% for ROM and 17.03% for BP. At 100°C the maximum relative error is about 16.05% in the case of ROM prediction and about 24.23% in the case of BP prediction. For the other temperatures the maximum relative error is 7.04% (ROM), 16.94% (BP) and 4.42% (ROM), 0.93% (BP) for 120°C and 140°C respectively.

In the case of tensile strength, the prediction quality is better than the elongation at rupture. In order to make clear this quality improvement, Table 2 shows the relative error for all aging temperatures and for both training algorithms. The relative maximum error reaches in the case of 80°C 5.05% for ROM prediction and 6.29% for BP prediction. At 100°C, the maximum relative error is 6.32% for ROM and 19.02% for BP. In the case of 120°C the quality of prediction is practically the same for BP (maximum error 4.53%) and ROM (maximum error 4.75%). At 140°C the relative error reaches 0.97% and 2.46% for ROM and BP respectively.

Table 1 Relative error of prediction in the case of elongation at rupture

Aging time (h)	Error (%)							
	80°C		100°C		120°C		140°C	
	ROM	BP	ROM	BP	ROM	BP	ROM	BP
0	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
500	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
1000	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
1500	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
2000	0,00	0,00	0,00	0,00	0,00	0,00	4,42	0,93
2500	3,55	15,52	11,89	21,64	1,25	16,57		
3000	5,87	17,03	16,05	24,23	3,14	4,94		
3500	2,52	12,32	9,05	13,50	7,04	16,94		
4000	10,00	14,75	13,64	13,80				
4500	14,81	16,37	4,37	4,26				
5000	2,58	3,00	7,91	4,06				

Table 2 Relative error of prediction in the case of tensile strength

Aging time (h)	Error (%)							
	80°C		100°C		120°C		140°C	
	ROM	BP	ROM	BP	ROM	BP	ROM	BP
0	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
500	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
1000	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
1500	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
2000	0,00	0,00	0,00	0,00	0,00	0,00	0,97	2,46
2500	0,41	3,67	4,84	5,94	4,74	1,52		
3000	3,28	1,64	6,32	7,06	4,62	0,81		
3500	5,05	2,23	0,69	1,92	2,49	4,53		
4000	0,77	0,78	5,89	19,02				
4500	0,47	2,25	5,27	13,40				
5000	0,33	6,29	6,01	14,01				

6 Conclusion

The prediction of accelerated thermal aging of cross-linked polyethylene properties using neural networks has been investigated in this study. The use of neural networks presents an alternative solution of the expensive and time consuming of laboratory experiments. Predicted and laboratory results are compared to validate the proposed neural networks. The proposed neural networks reproduce the same nonlinear characteristics obtained in laboratory with acceptable error. We have

pointed out that the NN trained by ROM is better than the NN trained by BP. The quality improvement of the NN trained with ROM is due to its ability to adjust the weights without determining any gradient. It had also a better ability in the generalization phase.

We may propose the application of this method to other kinds of aging (electrical, electrochemical...) and in general, in the cases where we need to determine the extrapolation of non-linear functions giving the variation of a property versus a given parameter at different constraint levels.

References

- [1] Ilie, S., Setnescu, R., Lungulescu, E.M., Marinescu, V., Ilie, D., Setnescu, T., Mares, G.: Investigations of a mechanically failed cable insulation used in indoor conditions. *Polym. Test* 30, 173–182 (2011)
- [2] Gulmine, J.V., Janissek, P.R., Heise, H.M., Akcelrud, L.: Degradation profile of polyethylene after artificial accelerated weathering. *Polym. Degrad. Stab.* 79, 385–397 (2003)
- [3] Tavares, A.C., Gulmine, J.V., Lepienski, C.M., Akcelrud, L.: The effect of accelerated aging on the surface mechanical properties of polyethylene. *Polym. Degrad. Stab.* 81, 367–373 (2003)
- [4] Fothergill, J.C., Dodd, S.J., Dissado, L.A.: The measurement of very low conductivity and dielectric loss in XLPE cables: a possible method to detect degradation due to thermal aging. *IEEE Trans. Dielect. Electr. Insul.* 18(5), 1545–1553 (2011)
- [5] Boukezzi, L.: The effect of thermal aging on the cross-linked polyethylene properties used in high voltage cables insulation. Dissertation, National Polytechnic School of Algiers (2007) (in French)
- [6] Boukezzi, L., Nedjar, M., Mokhnache, L., Lallouani, M., Boubakeur, A.: Thermal aging of cross-linked polyethylene. *Ann. Chim. Sci. Mat.* 31(5), 561–569 (2006)
- [7] Salama, M.M.A., Bartnikas, R.: Determination of neural-network topology for partial discharge pulse pattern recognition. *IEEE Trans. Neu. Net.* 13(2), 446–456 (2002)
- [8] Chen, H.C., Gu, F.C., Wang, M.H.: A novel extension neural network based partial discharge pattern recognition method for high-voltage power apparatus. *Exp. Sys. Appl.* 39, 3423–3431 (2012)
- [9] Mokhnache, L., Boubakeur, A.: Comparison of different back-propagation algorithms used in the diagnosis of transformer oil. In: *IEEE Ann. Rep. Conf. Electrical Insulation and Dielectric Phenomena (CEIDP)*, pp. 244–247 (2002)
- [10] Gençoğlu, M.T., Cebeci, M.: Investigation of pollution flashover on high voltage insulators using artificial neural network. *Exp. Sys. Appl.* 36, 7338–7345 (2009)
- [11] Khemaissia, S., Morris, M.S.: Review of networks and choice of radial basis function networks for system identification. *Technologies Avancées*, 55–85 (1994)
- [12] Mokhnache, L., Boubakeur, A.: Use of Neural Networks in the Monitoring of High Voltage Insulation Thermal Ageing. *J. Inter. Condi. Monit.* 6(4), 18–23 (2003)

Computer Simulation Analysis of Cluster Model of Totally-Connected Flows on the Chain Mail*

Alexander P. Buslaev and Pavel M. Strusinskiy

Moscow State Automobile and Road Technical University,
Leningradskiy Avenue 64, 125319, Moscow, Russia
apa12006@yandex.ru,
perssot@gmail.com

Abstract. This article considers a cluster model of totally-connected flows and also analysis software. This flow model is initiated by the relevance of creation of appropriate methods describing the traffic behavior by physical methods, Kerner, where there are some elements in implicit form. The model was first formulated at ITSC-2011, Bugaev etc, and other publications in Russia. It combines limit conditions of leader-following model, Lighthill-Whitham hydrodynamic approach and generalized solutions with Renkine-Hugoniot conditions. Experiments on the basis of each model are carried out, results are obtained and software is developed.

1 Introduction

1.1 *The Aim of the Article*

The aim of this article is simulation of cluster totally-connected flow model [2], [8], [9], based on solution of ordinary differential equations system (ODE) with variable number of components. Some components disappear at certain moments of time. It means that particles in these segments go into other segments with their density of flow. The goal is to find stationary solution, investigate qualitative properties, independently of initial conditions. In this work we consider two problems:

- 1) many clusters move on canonical supporters \mathbf{R} and \mathbf{T} ;
- 2) one cluster problem on the chain mails, which are systems of rings.

For example, the problem of chain mail flows like trains move on railroad or traffic flow on network. There is determination of sufficient conditions for the cluster length, where synergetic effect is available for any initial conditions. Traffic flow theory is considered as the main use [3], [4], [5], [6].

* This work has been supported by the Russian Foundation for Basic Research, grant No. 12-01-00794a.

1.2 Definition of Cluster

Cluster is modeled as geometrical figure, considered as a stable system of moving particles, distributed uniformly (chain of particles) (Fig. 1) in leader-following model [8]. These particles have equal velocity independent from maximum speed and distance between cars (density of traffic flow) [1]. Cluster measurements are derived from two components:

- 1) length of cluster d , or length of a lane, in meters (m);
- 2) height of cluster y , or density of traffic flow, in particles per meter (p/m).

M is number of particles in cluster or **mass** of the cluster:

$$M = d \cdot y \tag{1}$$

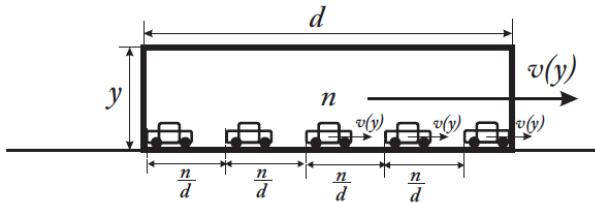


Fig. 1 Chain of particles as isolate cluster

Equations (3) and (6) are received from the law of conservation of mass.

Velocity v of cluster is described as monotonous function of its density: $v = f(y)$, in kilometers per hour (km/h). For example,

$$v(y) = v_0 \cdot \frac{y_{\max} - y}{y_{\max}} \tag{2}$$

When isolated cluster is moving, it's mass, density, length and velocity are equal to constant value.

Zero-cluster (Fig. 2) is a cluster with density, equal to 0: $y = 0$. Velocity of zero-cluster is equal to maximum speed: $v = v_0$.

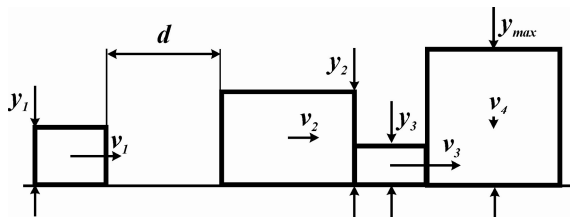


Fig. 2 Chain with Zero-cluster and Max-cluster

Max-cluster (Fig. 2) is a cluster with maximal density: y_{max} . Velocity of max-cluster is equal to 0: $v_d = 0$.

1.3 Classification of Flow Support

Lane (Fig. 3) is a segment **S** of a straight line **R** with direction of traffic flow and length d_0 . Clusters move on a lane one by one in the same direction.

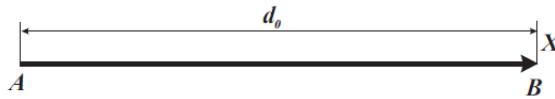


Fig. 3 Lane

Ring **T** is a segment **S**, where $A \equiv B$ (module d_0).

Infinite lane **R** (real lane) is limit of **S** with $A \rightarrow -\infty$, $B \rightarrow \infty$.

Chain of intersecting rings will be called Necklace. Every ring has two intersections P^1_i and P^2_i with previous and next neighbor rings. If the first and the last rings don't have intersections with each other, it will be called open Necklace. If they have intersection, it will be called close Necklace.

Chain mail is a system of n Necklaces (Fig. 4).

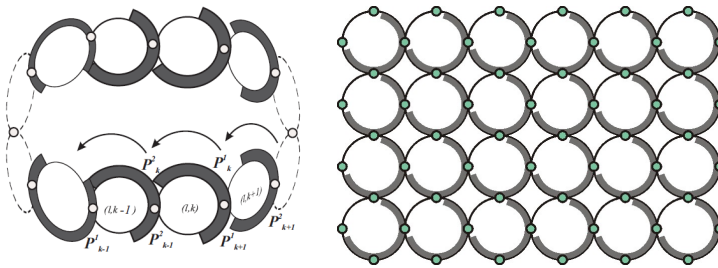


Fig. 4 Close Necklace and Chain mail

2 Cluster Flow on T and R

2.1 Description

Let us divide the flow of particles into segments, called clusters. The height of every cluster is equal to density y_i of the flow on each segment of the lane with length d_i , where density of flow is constant at a definite moment of time $t = 0$. Velocity of flow on every such segment is equal to v_i .

Clusters will interact with each other through boundaries with their neighbors. There are no gaps between clusters. If a slow cluster moves after a fast one, it will try to overtake the fast one. This is definition of clusters behavior model. The flow of particles inside the slow cluster will increase its velocity, entering the cluster with low density and vice-versa.

2.2 Cluster Flow Model for R

On Fig. 5 there is chain of clusters on real lane **R**. It is described by a system of equations (3).

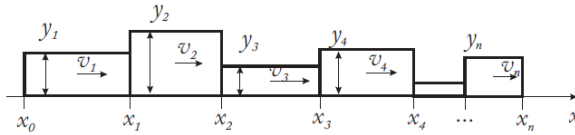


Fig. 5 Flow of clusters on **R**

$$\left\{ \begin{array}{l} \dot{x}_0 = v_1 \\ \dot{x}_1 = \frac{v_2 y_2 - v_1 y_1}{y_2 - y_1} \\ \dot{x}_2 = \frac{v_3 y_3 - v_2 y_2}{y_3 - y_2} \\ \dot{x}_3 = \frac{v_4 y_4 - v_3 y_3}{y_4 - y_3} \\ \dots\dots \\ \dot{x}_{n-1} = \frac{v_n y_n - v_{n-1} y_{n-1}}{y_n - y_{n-1}} \\ \dot{x}_n = v_n \end{array} \right. \quad (3)$$

In common, interaction coordinates between clusters are described by a system of non-linear ordinary differential equations (NODE). Cluster $[x_{n-1}, x_n]$ is the leader, and x_n is the front of this leader.

$$x_{00}=x_0(0), x_{10} = x_1(0), \dots, x_{n0} = x_n(0) \quad (4)$$

With initial conditions (4), system of equations (3) is the Cauchy problem. Besides, construction (3)-(4) supposes that

$$x_0(t) < x_1(t) < x_2(t) < \dots < x_n(t) \tag{5}$$

If at some moment of time one of inequalities (5) turns into equality, it means, that appropriate cluster disappears and the number of inequalities decreases by one unit.

The software, reconstructing interaction of clusters on **R**, was developed. By means of this software, experimental results, considered by that cluster model, were taken out.

At the beginning, this program saves initial conditions, defined by software user, by dynamic arrays or in arrays with variable size. Then, at every step, the model solves the system of differential equations with initial conditions (Cauchy problem) and constructs the flow of clusters on the lane and plots graphic of system solution.

The software was developed by:

- 1) environment development Microsoft Visual Studio 2010 Express;
- 2) program language C#;
- 3) .NET Framework 4.0.

2.3 Cluster Flow Model for T

On Fig. 6 there is chain of clusters on ring **T**. It is described by a system of equations (6).

The coordinate x_0 , at the interaction of clusters on ring **T**, is described by the same equation, as the coordinate x_n , because this type of movement of the first and the last clusters will interact, unlike clusters moving on the infinite lane.

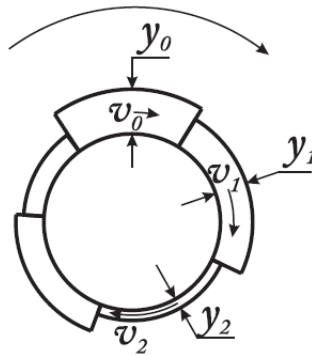


Fig. 6 Connected flow of clusters

$$\left\{ \begin{array}{l} \dot{x}_0 = \frac{v_1 y_1 - v_n y_n}{y_1 - y_n} \\ \dot{x}_1 = \frac{v_2 y_2 - v_1 y_1}{y_2 - y_1} \\ \dot{x}_2 = \frac{v_3 y_3 - v_2 y_2}{y_3 - y_2} \\ \dot{x}_3 = \frac{v_4 y_4 - v_3 y_3}{y_4 - y_3} \\ \dots\dots \\ \dot{x}_{n-1} = \frac{v_n y_n - v_{n-1} y_{n-1}}{y_n - y_{n-1}} \\ \dot{x}_n = \frac{v_1 y_1 - v_n y_n}{y_1 - y_n} \end{array} \right. \quad (6)$$

The developed software can reconstruct interactions of clusters on the ring. By means of this software, experimental results were taken out.

On Fig. 7 in lower part one can see curves of moving coordinates of boundaries between clusters.

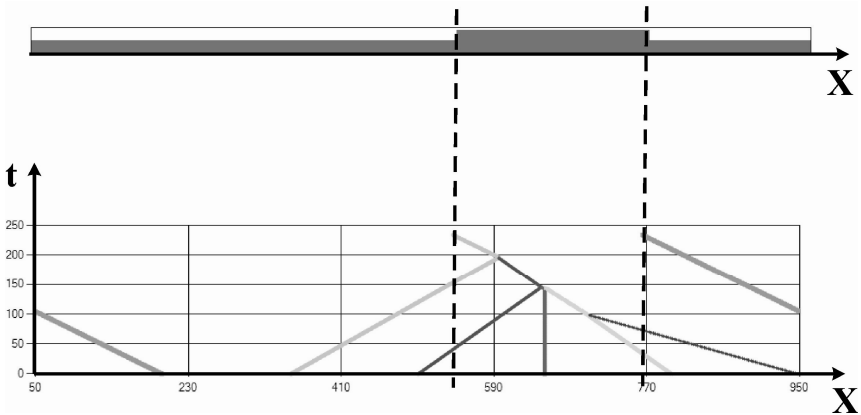


Fig. 7 Result of cluster's interaction on the ring

2.4 Results

- 1) During the interaction, the number of clusters doesn't increase on **R** and on **T**;
- 2) If the chain of clusters moves on the infinite lane, then by $t \rightarrow \infty$, with the measure zero of the initial conditions, the only one cluster will move, which has the number n at the start of modeling;

- 3) If the chain of clusters is moving on the ring, then by $t \rightarrow \infty$, with the measure zero of the initial conditions, the number of clusters will decrease as soon as there will be a permanent process of moving $2 \cdot k$ clusters with densities y_1 and y_2 (Fig. 8).

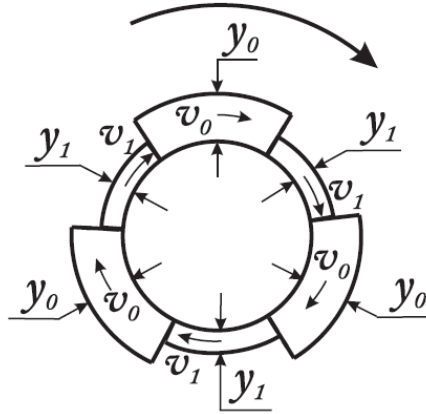


Fig. 8 Stationary process for T

3 Flow on Chain Mail

3.1 Rules

Let $2n$ be the number of crossing rings in the chain mail. On the rings there are $2n$ moving clusters, one on each ring. The length of each cluster is equal to l . The velocity of each cluster $v_i = v$ (Fig. 9).

The rules of passage through point P^1_i :

- 1) if the lane has the number $2i$, $v_{2i} = 0$; if P^1_i in $[x_{1_{2i-1}}, x_{2_{2i-1}}]$, otherwise $v_{2i} \neq 0$;
- 2) if the lane has the number $2i+1$, $v_{2i+1} = 0$; if P^1_i in $[x_{1_{2i+2}}, x_{2_{2i+2}}]$, otherwise $v_{2i+1} \neq 0$.

The rules of passage through P^2_i :

- 1) if the lane has the number $2i$, $v_{2i} = 0$; if P^2_i in $[x_{1_{2i+1}}, x_{2_{2i+1}}]$, otherwise $v_{2i} \neq 0$;
- 2) if the lane has the number $2i+1$, $v_{2i+1} = 0$; if P^2_i in $[x_{1_{2i}}, x_{2_{2i}}]$, otherwise $v_{2i+1} \neq 0$.

Average velocity of the whole system:

$$V_{av}(t) = \frac{1}{n} \sum_{k=1}^n v_k(t) \leq v \tag{7}$$

The length of each ring is 360 degrees. Intersections P^1_i and P^2_i are located at the interval of 180 degrees.

This model considers, that clusters cannot shrink, at the moment of waiting in points P_i . This means, that the flow of particles moves in organized chain without changing its density.

Developed software reconstructs cluster's movement on the chain mail with the given number of connected rings, length and density of clusters.

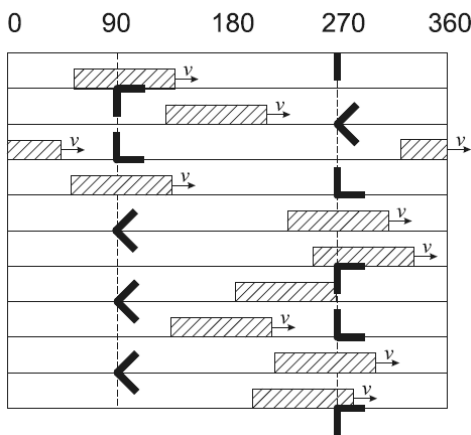


Fig. 9 Equivalent to Fig. 4

The software models the movement of clusters and constructs diagrams of the clusters velocities. The software also shows the average velocity of all moving clusters on the chain mail.

This software allows to determine initial values of the model in auto-mode, according to the random conditions, specified by model and software user.

3.2 Results

- 1) If the cluster length $l \leq 180$, for all initial system conditions, then this system adapts to maximum velocity of movement without stoppage (synergy);
- 2) If the sum of two neighboring clusters > 360 , for all initial system conditions, then velocity of the system $V_{av}(t) < v_i$, where v_i is velocity of i -cluster;
- 3) If the length $l > 180$, for all initial system conditions, then velocity of the system $V_{av}(t) = 0$ (Fig. 10).

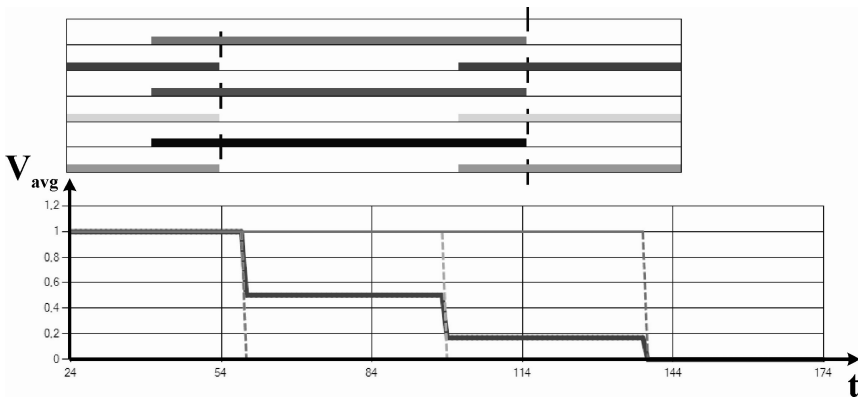


Fig. 10 Interruption of the system

4 Future Works

At present time the new software processing is in the development, which allows to reconstruct cluster's movement on the several lanes, on the belt and on the infinite belt.

Several lanes are a system of n parallel lanes, where each lane has the same direction and the same length d_0 . In this system clusters have ability to change neighboring lanes (Fig. 11).

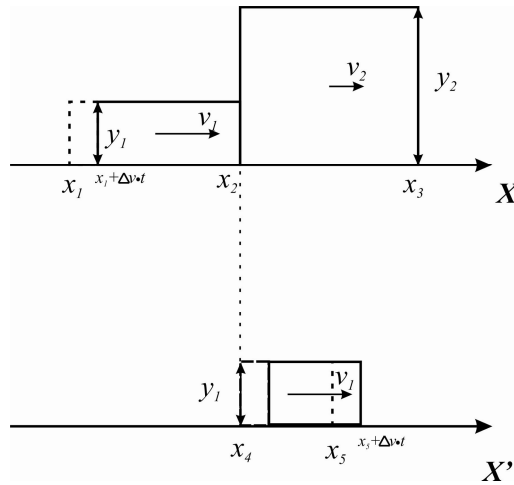


Fig. 11 Cluster changes a lane

Belt is a system of n rings with the same length (Fig. 12).

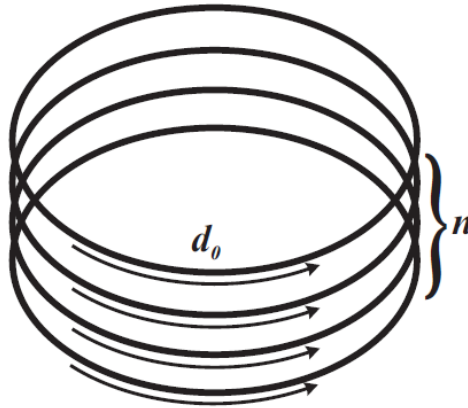


Fig. 12 Belt

Infinite belt is a system of n lanes, where each lane has the length equal to ∞ . The following problem statement is defined as:
let X and X' be two neighbor lanes for movement of clusters in the same direction from left to right. There are two clusters on lane X :

- 1) cluster, moving behind, is a fast cluster with density y_1 , velocity v_1 and length d_1 ;
- 2) cluster, moving ahead, is a slow cluster with density y_2 , velocity v_2 and length d_2 ;
- 3) fast cluster will overtake the slow one by a definite period of time Δt and coordinate x_2 of the fast cluster will be equal to coordinate x_3 of the slow one.

After that four strategies of cluster's behavior are considered:

- 4) fast cluster passes the slow one;
- 5) fast cluster integrates into slow cluster;
- 6) hybrid strategy of cluster's behavior;
- 7) fast cluster "waits behind" the slow one.

Analysis and verification of the hypothesis of the software development will follow.

5 Conclusion

In this work new software was created. By means of this software we obtained simulation results and proved theoretical and numerical results, that were made in [1], [2], [7], [8], [9], [11]. Now we can obtain some results of this model in created

software, which is very hard to obtain by theory and calculations, performing numerous experiments. Further we can explain these results by means of theory and computing.

References

- [1] Buslaev, A.P., Tatashev, A.G., Yashina, M.V.: Cluster flow models and properties of appropriate dynamic systems. *Journal of Applied Functional Analysis* 8, 54–76 (2012)
- [2] Kozlov, V.V., Buslaev, A.P., Tatashev, A.G.: Monotonic random walks and clusters flows on networks. In: *Models and Traffic Applications*, 300 p. Lambert Academic Publishing (2013)
- [3] Lighthill, M.J., Whitham, G.B.: On kinematic waves: Theory of traffic flow on long crowded roads. *Proc. R. Soc. London, Ser. A* (1955)
- [4] Nagel, K., Schreckenberg, M.: A cellular automation model for freeway traffic. *Phys. I France* 2 (1992)
- [5] Daganzo, C.F.: *Problem Sets: Fundamentals of Transportation and Traffic Operations*, Institute of Transportation Studies, University of California at Berkeley, 53 p. (1998)
- [6] Lukanin, V.N., Buslaev, A.P., Trofimenko, Y.V., Yashina, M.V.: *Traffic flows and environment*, Moscow, 408 p. (1998) (in Russian)
- [7] Buslaev A.P., Yashina M.V., Lebedev A.A.: Modeling of flows on the graphs. Theoretical and computing aspects. Part 1. NODE-model of traffic, Moscow, 105 p. (2011) (in Russian)
- [8] Kozlov, V.V., Buslaev, A.P.: Metropolis traffic modeling: from intelligent monitoring through physical representation to mathematical problems. In: *International Conference on Computational and Mathematical Methods in Science and Engineering*, vol. 1, pp. 750–756 (2012)
- [9] Buslaev, A.P., Yashina, M.V.: Cluster flow of total-connected flow with local information. In: *International Conference on Computational and Mathematical Methods in Science and Engineering (CMMSE)*, vol. 1, pp. 225–232 (2012)
- [10] Kerner, B.S.: *The physics of traffic*, 683 p. Springer (2004)
- [11] Bugaev, A.S., Buslaev, A.P., Kozlov, V.V., Yashina, M.V.: Distributed problems of monitoring and modern approaches to traffic modeling. In: *14th International IEEE Conference on Intelligent Transportation Systems (ITSC 2011)*, Washington, USA, pp. 477–481 (2011)

Assessment of Network Coding Mechanism for the Network Protocol Stack 802.15.4/6LoWPAN

Michał Byłak and Dariusz Laskowski

Military University of Technology, Warsaw,
00-908 Warsaw, Gen. S. Kaliski 2 Street
michal.bylak@gmail.com,
dlaskowski@wat.edu.pl

Abstract. The protocol stack 6LoWPAN (Low power Wireless Personal Area Networks) is minimally resources protocol that has been implemented in embedded operating systems. However, it improved mechanism for more efficient use of limited radio bandwidth. This mechanism is network coding, which is devoted to this article. The purpose of this article is to evaluate the effectiveness of network coding mechanism implemented on a network component that uses Atmel AVR Raven IEEE protocol stack 802.15.4/6LoWPAN. Implementation mechanism of network coding has been done in the environment and Contiki on the sample that is representative hardware platform Atmel AVR. As an interim step that would allow the implementation of the mechanism was to analyze the coding method for the network protocol stack 802.15.4/6LoWPAN.

1 Introduction

Wireless networks, especially the PAN (Personal Area Network) is characterized both by large losses of data during radio propagation and limitations associated with the available resources and the limited capacity of the radio channel. Meeting the requirements of the interoperability of IPv6 networks and ensuring a better use of the available radio bandwidth required to improve the existing protocol stacks for ad-hoc networks and in particular for sensor networks. One of the many communities it is open and flexible 6LoWPAN protocol (Low power Wireless Personal Area Networks). Resources of light-weight protocol, implemented in operating systems, you can improve the mechanism for the effective use of limited radio bandwidth. This mechanism is network coding, which is devoted to this article. Exactly, it can be said that the aim of this article is to evaluate the effectiveness of fermented mechanism implemented in the component network coding Atmel AVR Raven network protocol stack using IEEE 802.15.4/6LoWPAN. The implementation of network coding mechanism was built using a customizable system on the hardware platform Contiki Atmel AVR (RZ RAVEN).

2 Analysis of Network Coding Methods for the Network Protocol Stack 802.15.4/6LoWPAN

6LoWPAN implementation of IPv6 is tailored for the data link layer protocol based on the IEEE 802.15.4 standard designed for low-power wireless networks. Head in this protocol is based on standard IPv6 (RFC 2460) for size 40 B at the MTU (Maximum Transmission Unit) for the IEEE 802.15.4 standard ratio of 127 B. The use of smaller packets during transmission provides a lower packet loss ratio and stability data sharing. However, the low bandwidth radio link to the standard of 250 kbps forced to find ways to make better use of limited bandwidth. In the research project [4] is an example of the concept of network coding mechanism for the network layer packet (NC_NET) in tactical ad-hoc networks. Bringing this mechanism for IP packets in the example scenario with four nodes in Figure 1 shows.

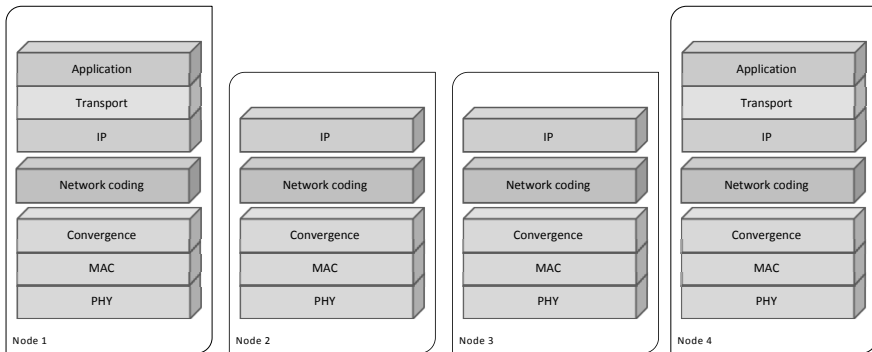


Fig. 1 Network coding range NC_NET

This mechanism takes into account similar size packages are subject to network coding operations. This is due to the fact that in the case of large differences in the size of each package must be a padding packet is less detrimental to the efficiency of coding. NC_NET mechanism consists of sub-mechanisms, namely:

- decoding buffer;
- recognition of the neighborhood;
- coding;
- confirmation of received packets.

3 Implementation of the Selected Network Coding Mechanism

The introduction of an appropriate system configuration consisting in modifying the source code helped prepare adequately implementing operating position. The result was the realization of multi-step with running component from intercepting

packets generated by each node. Intermediate stages of the commissioning tests included: running routing protocol, enabling confirmation ACK in IEEE 802.15.4, and change the addresses of each node.

With a view to implement the selected network coding mechanism adapted to networks with limited resources, it was necessary to take into account the resources available to the deployed storage network component. Figure 2 shows the resources occupied by the various protocols built into the system and what goes with it, the amount of memory available to implement a mechanism for network coding.

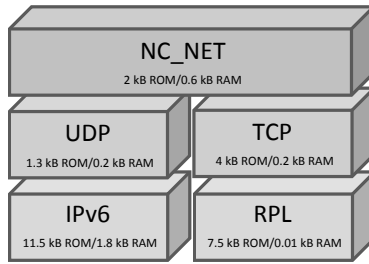


Fig. 2 Summary of memory resources used in the Contiki

4 The Implementation of the Commissioning Tests

In order to complete the commissioning tests were made in the structure of chain. It was based on a multi-step consisting of a representative system Contiki (Atmel AVR Raven - RZ RAVEN) equipped with an 8-bit microcontroller ATMe-gal284P (16 KB RAM and 128 KB flash). This system has a 2.4 GHz transceiver running accordance with the protocols ZigBee, IEEE 802.15.4, 6LoWPAN and RF4CE (Fig. 3).



Fig. 3 Kit Atmel AVR RZ Raven [12]

For testing commissioning and subsequent functional testing and quality served characterized test bed containing more than three terminals device and PC acting as (Fig.4), i.e.:

- The position of the compilation and implementation of the source code.
- The position of the analyzer package.
- The position of the communication event AVR RAVEN.



Fig. 4 Photo made a complete position to research

5 Conducting Qualitative Research

The purpose of qualitative research is to determine the number of the received data by the end node and specify the delay sent streams of data. In these studies take into account the following aspects such as:

- Holding time packets in the buffer mechanism of network coding;
- The Number of:
 - buffers used to encode the network nodes;
 - packets sent in a single data stream;
 - send streams of data units using ICMPv6;
- The interval between packets sent;
- The size of a single packet is sent.

Wireshark software using the planned research were captured packets from the position of the analyzer packages, which are then subjected to filtration and analysis allowed the preparation of the test results. Was taken into account the number of received data in one direction by the end node. Both packages were intercepted coded by network coding mechanism, as well as the unencrypted due to the buffer occupancy. For qualitative research, the following assumptions:

- packet size ICMPv6 Echo Request - 10 B;
- the number of packets in the data stream - 30;
- input variables:
 - the number of used buffer queues used network coding;
 - the number of streams of data - from 1 to 7.

During the study interval adjusted test packets, depending on the number of buffers used to hold packets and the number of data streams to bring non-memory resources overload situation and operational network component. Detention time packets in the buffer remained constant.

Test packets to send ICMPv6 Echo Request command uses the standard Ubuntu Linux terminal. To run at the same time a larger number of data streams used a simple script that allowed for accurate execution of qualitative research. In addition, results from the course of the operation ping were saved to individual text files.

Qualitative research results collected from Wireshark, which allowed the use of filtering the captured packets and initiates an option Statistics Summary of Statistics. In addition, this program has allowed checking the correct content generated packets containing encoded packets. Figure 5 presents the Wireshark window with the selected relevant information necessary to research.

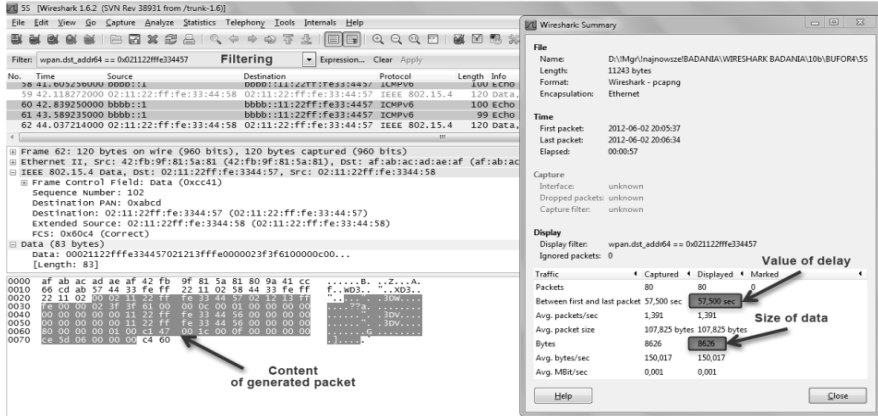


Fig. 5 Using Wireshark to retrieve the results of the qualitative research

It should be noted that, when setting the number of nodes in the network buffers for encoding a zero if there is no network coding mechanism used. In this way it is possible to determine the rate of profit in terms of growth coding volume of data useful and estimate the cost of the use of this mechanism, which is the estimated increase in packet delays. To determine the coefficient of profit network coding mechanism used in the relationship:

$$E_{cod} = \frac{D_1 - D_2}{D_1} \cdot 100\% \tag{1}$$

where:

- D_1 – the number of the received data by the end node without network coding,
- D_2 – the number of the received data by the end node using coding.

The negative results obtained $E_{cod} = 0\%$. This research applies only to the result of a data stream for which network coding mechanism has not been prepared. Based on the results plotted (Figure 6) presents the value of profit network coding mechanism depending on the number of used buffer queue and the number of data streams. When determining the value of delays are used as defined measure of cost encoding mechanism in the form of network delay. It means the amount by which the increased delay compared to a situation in which network coding mechanism has not been activated.

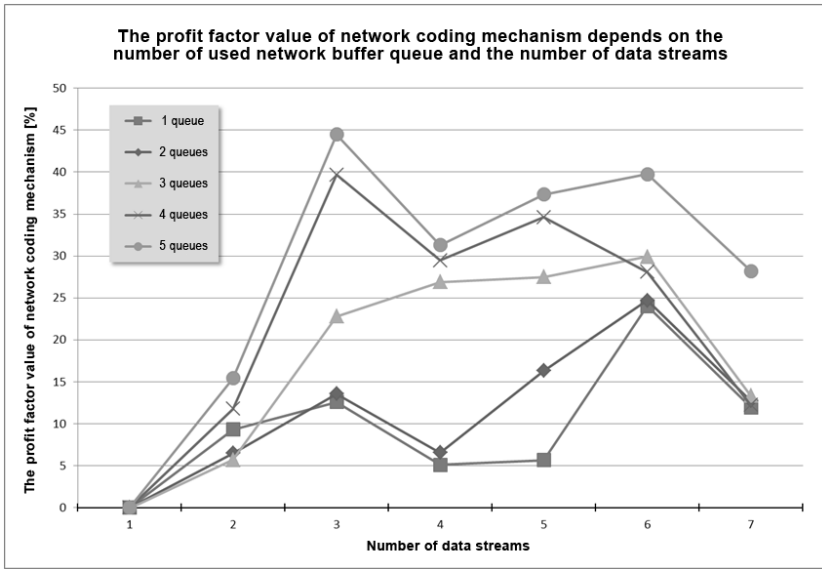


Fig. 6 Graph - value of profit network coding mechanism

Chart created from the results shown in the graph (fig. 7), which is dependent on the number of data streams and the number of used buffer queues.

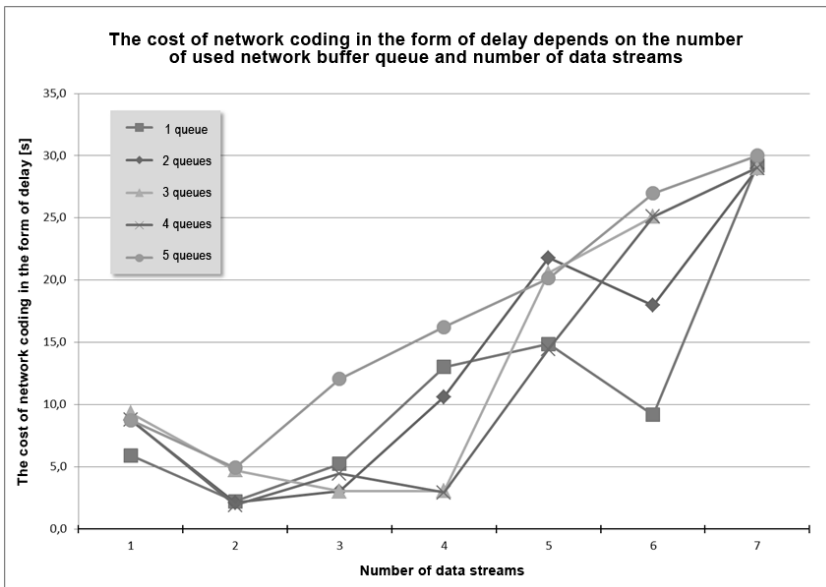


Fig. 7 Graph - the value of the cost of network coding mechanism in the form of delay

The results confirm that the number of received data in the terminal nodes decreases with the number of buffers diagnosis coding while keeping all packets sent from the testing station. This raises the profit of network coding at best, reaching up to the level of 44% with 5 buffers and sending three data streams. Only if we have a surplus of single stream of packets sent since the same source code and the delay made it does not allow the network coding gain. For this reason, defining the rate of profit network coding mechanism for negative values adopted a value of 0%.

The delay value is calculated between the first packet sent to a testing station last received a package in the terminal nodes for each test case and it is increasing with disabilities holding packets in the buffer and higher operating load and memory-network component, which implements the encoding mechanism. The smallest increase in the delay of received packets was 1.9 s, while the largest was 30 seconds. When analyzing the results of growth delay was defined name of the cost of network coding mechanism in the form of delay. The average value of this cost for all of the experiments was 13.5 seconds.

It should be noted that the time delay is dependent on the set interval between the sending of packets in the stream during the test quality. Setting an appropriate interval between packets allowed for correct operation of encoding mechanism. If you set too small intervals between packets in-time air sampling studies, this resulted in a temporary overload operating node and interrupt the operation of the network component.

6 Conclusions

In qualitative studies found that the mechanism implements network coding, all performed variations on change-buffer queues cotton bud number and the number of data streams, introduced an average coding gain of 17.9%, while the average delay increased by 13.5 second.

It should be noted that network coding mechanism operates only in the case of unicast packets addressed as for the study of the test package ICMPv6 Echo Request is appropriate. However, wanting to check the effectiveness of the coding for multicast and broadcast packets should have been implemented to improve the source code of network coding mechanism.

When analyzing the functionality of network coding mechanism implemented, do not forget about the impact of working memory load on the functioning of both the mechanism and the system built. Number of queue buffers used has a particular impact on the burden of memory resources. If you run a network component of the routing protocol set, and only the required server-settings (to run a simple web page) free memory is at the level of 79.4%. However, once implemented, network coding mechanism, this value increases to 80.8%. However, after each buffer value is increased by 1.2%. The result is that even with 5 buffers and a few commands into the console logging associated with the presentation of the contents of

buffers, free memory is around 90%. This load will have some problems such as with longer search paths in the routing table and the loss of a single packet.

Given, therefore, under the burden of memory and delay times, it is possible to determine the most efficient number of buffers in the implemented code. It turns out that 5 buffers already makes some problems with packet loss, so the best option seems to be set to 3 or 4 buffers taken packages. If the number of buffers faced the best nodes at number four data streams.

Noting the results obtained during the qualitative research it is possible to determine the application of this type of network coding mechanisms in practice. First of all, it can be used in situations where packet delay is not a priority in the data, but the important parameter is the desire to transfer large amounts of data. You can also implement packet detection mechanism, based on which node should decide whether specific data to be sent in the default mode or with the use of network coding mechanism. Definitely use of network coding mechanism (especially in sensor networks using network components with limited resources), aren't suitable for real-time data transmission. It may, however, be used for data transmission database, web and often sending data from different sensors, for example.

References

- [1] Katti, S., Rahul, H., Hu, W., Katabi, D., Médard, M., Crowcroft, J.: XORs in the air: Practical Wireless Network Coding. *IEEE/ACM Transactions on Networking* 16(3), 497–510 (2008)
- [2] Zhou, J., Xia, S., Jiang, Y., Zheng, H.: Decoding Buffer Management in Practical Wireless Network Coding. *IEEE* (2011)
- [3] Argyriou, A.: Wireless network coding with improved opportunistic listening. *IEEE Transactions on Wireless Communications* 8, 2014–2023 (2009)
- [4] Krygier, J.: Zaawansowane metody i techniki sterowania ruchem w taktycznych sieciach adhoc. Projekt badawczy własny (MNIS) nr O N517 274839, Warszawa (2011)
- [5] Krygier, J.: Realizacja środowiska implementacyjnego oraz wykonanie wstępnej wersji oprogramowania mechanizmów kodowania sieciowego. Projekt badawczy własny (MNIS) nr O N517 274839, Warszawa (2012)
- [6] Shelby, Z., Bormann, C.: *6LowPAN: The Wireless Embedded Internet*. Wiley (2009)
- [7] Kerningham, B.W., Ritchie, D.M.: *Język ANSI C*. Wydanie IX, WNT, Warszawa (2004)
- [8] King, K.N.: *Język C. Nowoczesne programowanie*. Wydanie II, Helion, Gliwice (2011)
- [9] IEEE Std 802.15.4-2006: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). *IEEE* (2006)
- [10] RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks. *IETF* (2007)
- [11] RFC 4919: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs). *IETF* (2009)
- [12] Official site of Atmel AVR Raven,
<http://www.atmel.com/tools/avrraven.aspx>

Reliability Analysis of Discrete Transportation Systems Using Critical States

Dariusz Caban and Tomasz Walkowiak

Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27,
50-320 Wrocław, Poland
{dariusz.caban,tomasz.walkowiak}@pwr.wroc.pl

Abstract. The paper presents a resource constrained model of a discrete transportation system that can be used to simulate its operation in presence of faults. The simulation results are used to determine the initial level of resources that ensures seamless operation of the system. The simulator is also used to assess the conditional probability of system failure after reaching a specific set of reliability states. This is used to determine the set of critical states, i.e. the states when the system is still operational but the probability of failure in near future is unacceptably high. These states can be used as an indicator that the system is degrading dangerously.

1 Introduction

Multistate approach is often used in reliability analysis [2, 3], mainly to reflect the partial system degradation as opposed to complete failure. The same approach can be used in ternary state analysis, where the third (critical) state corresponds to the situation that the system is dangerously degraded, and urgently needs maintenance action [1]. This critical state may be used to monitor the health of the system, provided that it does not generate too many false alarms.

In the paper, we consider the risks connected with managing a discrete transportation system (DTS), i.e. a transportation system in which goods are transported by a fleet of vehicles of limited capacity. The vehicles operate according to a fixed schedule, carrying goods between destinations. The goods are fixed in size (the volume of goods is always a discrete number). The proposed reliability analysis is based on the assessment of system operation using Monte Carlo simulation. A custom designed simulator is used [6], which utilizes the publicly available SSF simulation engine [4]. The simulator is used to establish a secure level of system resources (vehicles and drivers), to assess the probability of the system reaching a critical state of operation, and of the conditional probability of system failure once it reaches the critical state.

2 Discrete Transportation Systems Model

The paper considers the risks connected with managing a discrete transportation system (DTS), i.e. a transportation system in which goods are transported by a fleet of vehicles of limited capacity. The system consists of nodes (locations from which goods are collected and to which they are carried) and vehicles travelling between the nodes. The vehicles are manned by the human drivers. The system is modeled by the various interacting resources used to achieve the transportation goals.

2.1 Transportation Systems Resources

The considered system model is composed of [5]:

- the set of nodes X , in which a central node x_0 is distinguished (the local distribution center),
- the set of routes between the nodes R ,
- the set of vehicles V ,
- the set of transportation assignments Z ,
- the set of travel schedules C ,
- the set of vehicle operators (drivers) K , assigned to vehicles when they transport goods between the nodes,
- the set of maintenance teams M that service the vehicles after a break down.

Transportation Assignments

The assignments are connected with specific needs for transporting goods. It is expressed as a discrete number of standard containers. The assignments specify the source node from which the goods are picked and the destination node where they are carried to. The assignments are always either from the central node or to the central node. Assignments between other nodes are not allowed.

There is a fixed time in which each assignment must be completed. Depending on the nature of the DTS system, this time is fixed by local regulations or is part of the service agreement between the assignees and the transport service provider.

Nodes

There is a single central node and a number of local ones. The central node is the only destination of assignments generated at the local nodes. The central node generates goods assignments destined to all the local nodes.

The assignments are generated independently of each other, using random distributions. We use Poisson distribution for this purpose. Each local node has an attribute which determines its characteristic rate of assignments generation. The central node is described with an array of assignments rates, one for to each local node.

Vehicles

It is assumed that all vehicles of the same type have similar properties. They are described with the same functional and reliability related parameters: capacity (expressed as the number of standard containers), average cruising speed (determining the route latency), failure rate, renewal time. All the vehicles are based in the central node and travel from it to realize the assignments.

At any moment in time, a vehicle may be in one of the following states: it might be en route between nodes (a specific distance from the starting node, carrying specified amount of goods), it might be waiting for goods in a node, it might be stopped due to unavailability of a driver or due to regulatory rest period of its driver.

A vehicle may be realizing multiple assignments at the same time. It will be fully loaded with goods if the pending assignments allow it. If there are insufficient assignments for nodes towards which the vehicle is destined, then it may be partially loaded or even travelling empty. It collects goods en route if there are pending assignments in the visited nodes.

Vehicle Operators

The drivers are a limited resource of the system. If a vehicle is assigned to a job (to a timetable), a driver must also be associated to it. Any unallocated driver can be associated with any vehicle. Only one driver at a time is associated with a vehicle (since we do not consider long distance routes with standby drivers).

The work of vehicle operators is regulated by local and EU legislature. The daily working hours are limited (to 8 hours), there are also compulsory rest breaks while driving. Thus, at any time the driver can be in one of the following states:

- resting (not at work),
- available (at work – ready to start driving),
- pausing (having a break while driving),
- driving.

It is assumed that the drivers work in 8 hour shifts. The state of each driver changes to “resting” whenever his daily working time limit is exceeded and he arrives at the central node. He stays in this state until the beginning of his shift next day. Then, his state changes to available. If there is a pending driving schedule (timetable) and an available vehicle, then his state changes to “driving”.

While driving, the driver has to heed the limits on the maximum length of time that he can work without a break. Normally, the timetables assure that the required breaks are fulfilled while the vehicle is loaded in the visited nodes. If a route is unnaturally long or there are travel delays on the way, then the driver is required to take a break en route. The parameters determining the daily working hours limit, maximum uninterrupted driving period, minimum break duration are associated with the operators model.

The allocation of drivers to the jobs (described by the timetables in the model) is governed by some simple rules:

- Vehicles cannot carry goods between nodes if there is no operator available.
- The driver is chosen from among those, whose daily working time limit allows them to complete the job with at most 10% overtime (i.e. estimated journey time is less than 110% of the left work time limit).

Routes

Routes represent the direct connections between nodes of the system. They are characterized by the distance that the vehicles must travel. Taking into account the average travelling speed of vehicles, this determines the latency connected with moving from one node to another. This latency is further distorted by the travel delays which represent the natural variation of latency, e.g. caused by the traffic congestion. These delays are modeled using a random distribution.

Timetables

Vehicles are travelling in accordance to fixed timetables (travel schedules). Each timetable determines the time to leave the central node and a sequence of nodes that must be visited by the vehicle as well as the times of these visits. It describes the daily work of the vehicle associated with the timetable, independent of the actual needs as determined by the assignments.

Vehicles are loaded to their capacity (if there are sufficient assignments) at the central node on starting a travel schedule. On reaching each consecutive node in the timetable, the goods destined to it are unloaded and the goods waiting there are loaded in their place. The time used for unloading and loading is randomly chosen. If there are other vehicles in the node, then they are queued and the period of loading/unloading is extended commensurately. The timetables do not specify the time to leave a node (except the timetable start time).

When the vehicle returns to the central node (at the end of a schedule) it is completely unloaded. It can then be associated with another timetable or it may be placed in the pool of available vehicles, waiting to be associated with a job.

The timetables are not directly associated with vehicles or drivers. Instead, any available vehicle and operator is allocated to each schedule. If there are no vehicles or drivers available, then the timetable cannot be realized.

Maintenance Teams

The model does not distinguish any specific parameters of the maintenance teams, just their number. If a vehicle breaks down, it will be repaired by one of the maintenance teams. The distribution of the repair time is associated with the vehicle, not with the team.

2.2 Operational Faults

The following faults are taken into consideration during the system analysis:

- vehicle breakdowns (requiring repair by one of the maintenance teams),
- driver absentees at work (due to illness or other incidental leave),
- traffic congestion (which result in random delays in the timetables).

The vehicles are assumed to break down occasionally, in accordance to their reliability parameters (failure rates). They then stop operation and wait for a maintenance team. On being repaired (after a random repair time), the vehicles continue the work they were realizing before breakdown. No transloading of the goods is allowed. Each maintenance team repairs only one vehicle at a time. If all the maintenance teams are currently occupied, then the vehicle repair is delayed until one becomes available.

Drivers are liable to sickness and other events that can make them temporarily unavailable. After a prescribed leave of absence they come back to work. Driver illness is modeled as a stochastic process with three categories of illness [6]:

- short sickness (1 to 3 days),
- typical illness (7 to 10 days),
- protracted disability (10 to 300 days).

In the model the three types of disabilities are independently generated. In each case the actual period of absence is randomly chosen from the appropriate range.

Traffic congestion is not explicitly modeled as a fault. The delays caused by the traffic jams are considered as a random component of the route cruise times.

2.3 Simulation Technique Used to Analyze the System Model

The analysis is performed using a simulator, custom designed for this purpose [5,6]. It is based on the publicly available SSF simulation engine that provides all the required simulation primitives and frameworks, as well as a convenient modeling language DML [4] for inputting all the system model parameters.

By repeating the simulator runs multiple times using the same model parameters, we obtain several independent realizations of the same process (the results differ, since the system model is not deterministic). These are used to build the probabilistic distribution of the results, especially the average measures.

3 Critical States of Operation

The reliability state of the system is characterized by the number of operational resources, at a specific point in time. Thus, it is a vector of the number of drivers n_{K_i} , that are not ill, and the number of vehicles n_{V_i} that are operational:

$$S_i = [n_{K_i}, n_{V_i}]. \quad (1)$$

Initially, all the drivers and all the vehicles are available. This initial state of operation $S_o = [n_{K_o}, n_{V_o}]$ is a strategic system investment decision. It impacts both the reliability and the economic aspects of the DTS. It reflects the total resources committed to the system:

$$\begin{aligned} n_{K_o} &= \text{card}(K), \\ n_{V_o} &= \text{card}(V), \end{aligned} \tag{2}$$

where the sets K, V are defined in Section 2.1.

The quality of the system is assessed by its ability to transport all the goods on time, regardless of the reliability state. It is assumed that impaired operation cannot cause rejection of transportation assignments. Thus, all the assignments are serviced by the system, though delays in delivery may occur. The likelihood of these delays varies with the reliability state. This is the basis of classification of the states as operational, failed or critically operational (a border case between operational and failed).

3.1 Quality of System Performance

Each assignment has a guaranteed time of delivery Δt_g . The real time of delivery Δt is a random variable, which depends on the current volume of assignments, travel delays, reliability state, etc. There are two possible relations between the delivery deadline Δt_g and the actual assignment realization Δt :

- If the assignment is completed before the deadline, i.e. $\Delta t \leq \Delta t_g$, there is no penalized delay. There is no reward for the early delivery, either.
- If the assignment is completed after its deadline, $\Delta t > \Delta t_g$, then there is a late delivery penalty incurred.

The short term measure of the quality of performance is obtained by counting the assignments that are delivered on time (before the deadline). If the system is fully operational, it should realize almost all the assignments on time. If a fault occurs, some of the assignments are realized late.

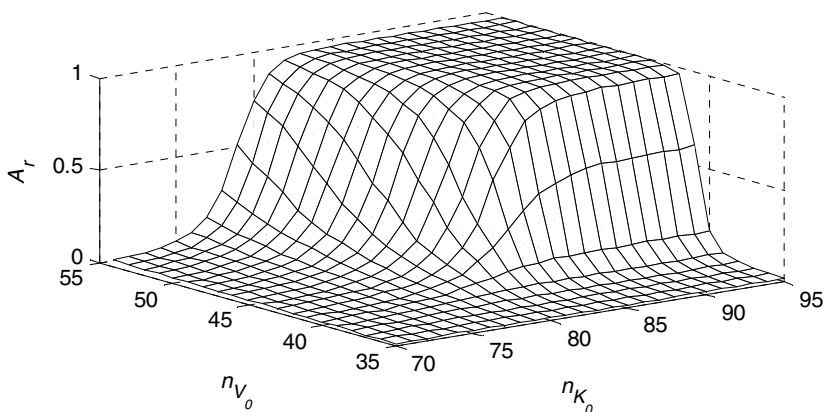


Fig. 1 The average ratio of on-time deliveries for various numbers of vehicles and drivers S_o

The ratio of on-time deliveries a_r is defined as the proportion of assignments that are delivered on time to the total number of assignments in the system during a fixed time period. Of course, this measure is a random variable that reflects the nondeterministic properties of the whole system. A 24 hour reporting period is assumed for determining this ratio.

The sequence of time instances (t_0, t_1, \dots, t_n) fixes the boundaries of the consecutive days, for which the ratio is considered. $N_d(t_i, t_{i+1})$ denotes the number of assignments completed in (t_i, t_{i+1}) . $N_{pd}(t_i, t_{i+1})$ denotes the number of those assignments, which are completed on time. Average ratio of on-time deliveries is defined as:

$$a_r(t_i) = \frac{N_{pd}(t_{i-1}, t_i)}{N_d(t_{i-1}, t_i) + 1} \tag{3}$$

The average ratio A_r , calculated over the various reporting periods and the various random variable realizations, is used to characterize the overall system performance:

$$A_r = E\{a_r(t_i)\} \tag{4}$$

The value of this ratio can be predicted using the simulation technique mentioned previously. Fig. 1 presents the results of such assessment for the system with different levels of allocated resources, as defined in (2).

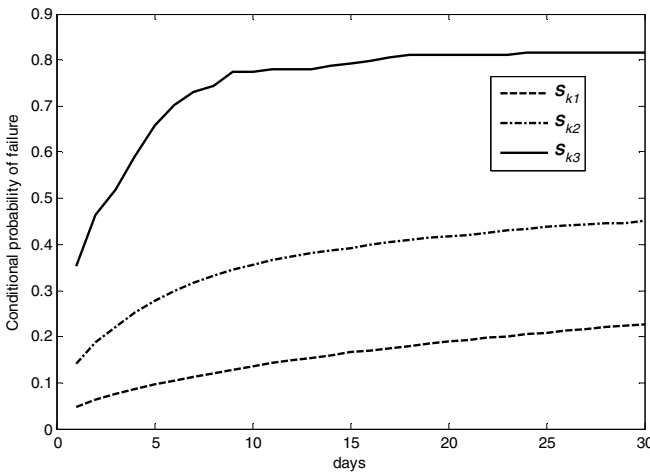


Fig. 2 The conditional probability of system failure for various time horizons and for variously defined critical states ($S_{k1} \supset S_{k2} \supset S_{k3}$)

3.2 Critical States

During system operation the actual number of available vehicles and drivers varies in time due to the operational faults. This is characterized by the reliability state, as defined in (1), at any given moment of time.

For any set of these states, it is possible to compute the conditional probability that in a short time horizon τ the average ratio of on time deliveries a_r does not drop below an acceptable level A_{crit} . The condition in this case is that at the beginning of the time period the system enters one of the states in the considered set.

This probability reflects the likelihood that attaining the considered set of states results in system failure. It is used to classify the reliability states into two disjoint sets – the set of operational states when the probability of failure is very small and the set of fail states if the probability is reasonably high. Of course, this classification depends on the level of probability that we consider “small” or “reasonably high”, as well as on the time horizon (a few days).

The critical states are defined as a border case between the operational and fail states [1], i.e. they are a subset of the operational states with just a single or a few drivers/vehicles more than in the fail state. They are used to predict impending system failure.

Fig. 2 illustrates how the conditional probability of failure changes depending on the value of time horizon τ for various definitions of the critical states. S_{k1} corresponds to a liberal definition of the set of critical states (with a few redundant vehicles and drivers), S_{k2} is a subset of S_{k1} with less redundancy, $S_{k3} \subset S_{k2}$ is the narrowest margin. It should be noted that a smaller critical states set yields a more likely prediction of system failure in a short time, but at a cost – the probability of its occurrence is smaller (in the considered example 0.437, 0.045 and 0.002 respectively).

4 Case Study

All the simulation results that illustrate the presented considerations (Fig.1 and 2) were computed using a real-life example of a transportation system. The system consists of a central node located in Wroclaw and 22 local nodes located throughout the region. The stream of assignments (generation of cargo) is assumed the same for all the destinations. It is modeled as a Poisson stream with the rate set to 4.16 per hour in each direction. On average this corresponds to 4400 containers to be transported every day.

The set of vehicles is uniform, each can carry 10 containers at a time. The velocity of vehicles is modeled by the Gaussian distribution (average of 50 km/h, standard deviation of 5 km/h). The average loading time is 5 minutes. The mean time to failure of each vehicle is assumed as 20,000 hours. The average repair time is 36 hours (left-truncated normal distribution with std. deviation of 18 hours).

The vehicles are operated by drivers working in 2 shifts (morning: 6 a.m. till 2 p.m., afternoon: from 1 p.m. until 9 p.m.). The rates of drivers illness for defined three categories are equal to: 0.003, 0.001 and 0.00025.

The system works with fixed timetables. These are organized so that the vehicles and drivers have a grace time of 20 minutes after completing one journey, before starting on the next one.

The initial numbers of vehicles and drivers are established on the basis of simulation results presented in Fig. 1. These numbers were fixed as 48 vehicles and 85 drivers, which ensure that the total volume of cargo, that can be transported daily, exceeds the average demand by 15% (overall). This is accepted as a reasonable level of redundancy in the discussed system.

5 Conclusions

The presented results bear out the feasibility of using the concept of critical state operation in the considered class of DTS systems. The choice of the set of critical states is arbitrary – the larger the set of operational states that we deem critical, the worse its predictive value (as illustrated in Fig. 2). On the other hand, the critical state is also more likely to occur. As such, it is a management decision that will influence the balance between trustworthiness of the predictions and unjustified alarms.

In case of a stable system, such as analyzed in the case study, the best predictions (represented by critical states set S_{k2}) are not particularly significant. The real value of the approach can be appreciated if we consider a system that may deteriorate from its model parameters, e.g. if the frequency of goods assignments increase unpredictably or the travel times increase due to deteriorating traffic conditions. Then, the occurrence of the critical states operation may be the first indication of the changes, raising alarm for the system management that a reorganization of the timetables is required.

Acknowledgement. The presented work was funded by the Polish National Science Centre under grant no. N N509 496238.

References

- [1] Kołowrocki, K., Soszyńska-Budny, J.: Integrated Safety and Reliability Decision Support System. *Journal of KONBiN* 4(20) (2011), doi:10.2478/v10040-012-0033-5
- [2] Lisnianski, A., Frenkel, I., Ding, Y.: *Multi-state System Reliability Analysis and Optimization for Engineers and Industrial Managers*. Springer (2010) ISBN: 978-1-84996-320-6
- [3] Natvig, B.: *Multistate Systems Reliability Theory with Applications*. Wiley Series in Probability and Statistics. Wiley, New York (2010) ISBN: 978-0-470-69750-4

- [4] Nicol, D., Liu, J., Liljenstam, M., Guanhua, Y.: Simulation of Large Scale Networks Using SSF. In: Proceedings of the 2003 Winter Simulation Conference, pp. 650–657 (2003)
- [5] Walkowiak, T., Mazurkiewicz, J.: Analysis of Critical Situations in Discrete Transport Systems. In: Proceedings of International Conference on Dependability of Computer Systems, Brunow, June 30-July 2, pp. 364–371. IEEE Computer Society Press, Los Alamitos (2009)
- [6] Walkowiak, T., Mazurkiewicz, J.: Human resource influence on dependability of discrete transportation systems. In: Zamojski, W., Kacprzyk, J., Mazurkiewicz, J., Sugier, J., Walkowiak, T. (eds.) Dependable Computer Systems. AISC, vol. 97, pp. 271–283. Springer, Heidelberg (2011)

A Reference Model for the Selection of Open Source Tools for Requirements Management

Bartosz Chrabski¹ and Cezary Orłowski²

¹ Gdańsk University of Technology, IBM Polska,

1 Sierpnia 8 bud A, 02-134 Warszawa

bartosz.chrabski@pl.ibm.com

² Prof. Gdańsk University of Technology,

ul. Narutowicza 11/12, 80-233 Gdańsk

cor@zie.pg.gda.pl

Abstract. The aim of this study is to build a reference model for the selection of Open Source tools for the management of customer requirements in IT projects. The construction of the reference model results from the needs of companies producing software which are also interested in streamlining the process of managing requirements using Open Source tools. This interest in Open Source tools is, in turn, a consequence of licensing costs, integration with the rest of the portfolio, and support costs. The advantage of Open Source tools is their low license cost and the ease of their adaptation, provided that there is access to a reference model for their adaptation. The problem of the IT market is the lack of such reference models for selecting Open Source tools. Therefore, the authors undertook to build such a model and to apply it in supporting the requirements development process in IT projects.

To achieve the objective, the study was divided into four main parts. The first elaborates on the issue of selecting tools in the software development cycle, indicating the need for departments creating IT systems to use appropriate tools for the given organization. The second part is devoted to the approach to the selection of tools supporting the requirements development process. The purpose of this section is to diagnose the state of IT projects and the lack of support for the requirements development process. The third (the main) part presents the idea to construct a reference model for the selection of tools to support the requirements development process. The structure and development prospects of the model are also discussed here. The fourth part is entirely devoted to examples of the application of the reference model in several IT projects.

1 The Issue of Selecting IT Tools

Best practices, management and production methods as well as appropriate tools are essential for the development of the requirements management process for a particular organization. The choice of appropriate best practices, management methods or tools is a challenge and a problem for many organizations [8] [1].

Ideally, in the selection of tools, duplication of functionality between the various tools used or the inability to integrate data from different areas of software development does not occur. If we look at the current software development environment, we can clearly say that there is an abundance of manufacturing processes, and thus an inability to efficiently share information between the roles and tools. This is often combined with the selection of tools convenient for one group of stakeholders, which will disallow the development of the strategy for another department responsible for creating the systems. Analyzing best practices of the market, such as CMMI (Capability Maturity Model Integration), leads to a clear conclusion that we should strive to combine disciplines in the manufacturing process. This described process of integration should take place at the level of data repositories, without any further need for unnecessary duplication of core functions and information on which they operate.

The observations of the authors concerning the lack of information exchange between the stages of the manufacturing process were confirmed by a recent survey on the use of tools in the software life cycle. They clearly show that building a complete and integrated tool environment that will efficiently support all the processes is a challenge for organizations, which further prevents them from managing software development in a predictable way. The survey carried out by IBM at the Rational Innovate 2012 Conference [8] showed that the adaptation of tools to a client's needs and their integration is the biggest challenge for organizations. The results revealed the most common problems for organizations developing solutions:

- *tools which are easier to use, to get to know, and to adapt to needs – 82 %,*
- *tool optimization for large implementations, and adaptation for use by teams in companies – 57 %,*
- *strengthening tool integration in the software life cycle – 44 %,*
- *better reflection of the software life cycle in the organization – 20 %,*
- *tools that help standardize the processes of development and maintenance – 13 %.*

The introduction of various new tools in an organization, without an appropriate pre-planned strategy for their selection, quickly leads to a situation similar to that which has occurred in integrating large enterprise systems. A service-oriented architecture was the solution to the problems of system integration and cooperation and did not generate unnecessary costs [4], and it is this concept which constitutes the basis for further discussion. The authors assumed that analogically to service-oriented solutions, tools in the manufacturing process should be selected on the basis of the functional requirements of organizations and their cooperation through integration to increase data reuse.

In this article, the authors refer to the previously discussed challenges and try to answer the question of how to select tools to manage requirements so that they would support the process in which the organization operates and how to set strategies for the further evolution of development departments. The basic element

used in the further discussion is a reference model for tools to support the processes of requirements engineering, which can be used for proper selection, comparison and verification of the possibilities of integration.

2 Processes of Requirements Development and Their Support

The development of a model solution, such as a reference model for the selection of analytical tools, is not possible without a detailed examination of the software development processes as well as the discipline of developing and managing requirements. The branch of requirements engineering has for many years been the biggest challenge for project teams and people directly involved in working on them who are looking for a solution by trying to use different tools to support this process. Requirements are defined as a set of standards and rules which the system must meet, which has always played the most important role in IT projects, while being the primary criterion of assuring quality [3] [13] [16]. The very definition of requirements engineering states that its actions will be closely linked with other areas of the software development process, such as testing, change management and architecture management.

The branch has been divided into two areas called requirements development and requirements management. The first of these processes relates to collecting, storing, analyzing and validating requirements, and describes the conversion of business needs of stakeholders into project requirements [6]. Requirements management is a continuation of the requirements development process, additionally including the issues of analysis, tracking changes and progress, prioritization, communication, maintenance or change management [6]. As the characteristics of the branch show, it may not be possible to address all the needs while using only one of the available tools, and not all of the activities might be required by a customer in the currently used processes. It is a challenge for manufacturing-oriented organizations to select such tools in the area of requirements engineering which will accurately reflect their business needs.

The result of research conducted in 2009 by the Standish Group (CHAOS Reports) on the factors which influence the questionability of projects, were indicated as a lack of input from the user ~ 13%, incomplete requirements and specifications ~ 12%, and changes in requirements and specifications ~ 12%. According to data from the Standish Group, one of the three main factors that characterize a successful venture, in addition to user involvement and management support, is having clearly defined requirements [5] [15].

An analogy to the findings of the report can be found in even the most mature organizations where it is hard to get away from the challenges of requirements and their direct impact on the cost, quality and duration of a project. Increasing the time dedicated to a proper requirements analysis can contribute to substantial savings as demonstrated by the analysis of such projects within the NASA group. Table 1 presents a fragment of a report published by NASA, which relates to research projects and the impact of analysis on their direct progress.

Table 1 Cost overruns in the analyzed projects of the NASA group [7]

Project time devoted to analysis	Number of projects	Cost overrun
less than 5 %	5	125%
from 5 % to 10 %	7	83%
more than 10 %	6	30%

In view of the discussed problems, organizations increasingly realize that they can not succeed without appropriately addressing their needs and without good management. Despite a rich set of practices and methodologies to support requirements engineering, in practice, the discipline is not fully integrated with the processes of software development [2] [12]. The observed challenges result from the lack of dedicated tools that adequately support the processes of analysis or they were badly chosen in terms of the needs of the target group or the organization itself. The aforementioned behavior is often reflected in work on requirements in the form of documents which have no direct ability to integrate with solutions dedicated to supporting other disciplines of the production cycle. Propagating data manually between systems can lead to inconsistencies and unnecessary duplication.

On the basis of the analysis of the requirements engineering branch and the challenges presented earlier, the authors clearly stated that proper implementation is not possible without the adequate support of dedicated tools. As a result of not using the appropriate tools, such tasks as the complete realization of the business aims of a project or improving the manufacturing process itself are not possible. When analyzing the Open Source tools market, the authors concluded that it is particularly immature in the case of requirements engineering, and no clear direction for its development can be observed. This then prevents a discussion on best practices and full support for these processes. An analysis was carried out on COTS (Commercial Off-the-shelf) tools, based on experience with Open Source environments, which can serve as a model for developing free solutions and further inference.

The choice of commercial tools was dictated by their continuous development, technical support, documentation and support capabilities in the area of implementation services. While constructing the concept of a reference model, several solutions meeting these conditions and implementing an extensive range of functionalities were analyzed. The provider's contribution to the development of the branch, through innovativeness or the number of publications on requirements engineering, was not without significance in the analysis. While choosing the tools provider, the authors looked into the reports of Gartner and Ovum analytical and consultancy group on the integrated cycle of software development management [14]. Studies from 2008, 2010 and 2012 allowed focus on a group of IBM Rational products.

	RATING				
	Strong Negative	Caution	Promising	Positive	Strong Positive
Aldon			x		
Borland				x	
CollabNet			x		
IBM					x
Kovair			x		
Microsoft				x	
MKS				x	
Polarion			x		
Rally Software				x	
Serena Software				x	
TechExcel				x	
VersionOne			x		

As of 11 December 2008

Source: Gartner (December 2008)

Fig. 1 Summary of ratings for tools to support software development management (according to Gartner Inc.)

On the basis of the submitted reports by Gartner and Ovum, two of the most dynamically developing products from requirements engineering were selected from IBM's portfolio and were subjected to a functional analysis supplemented by support for best practices. The group included the Rational RequisitePro tool for use in more formalized projects, and Rational Requirements Composer often used in agile projects. These solutions were used as a norm in the construction of the reference model for requirements engineering in terms of functionality, technology support and integration. Here is a brief description of the IBM Rational tools focusing on their essential features relevant for building a reference model. The specific use of each tool for projects with varying degrees of formality has been clearly defined.

IBM Rational Requirements Composer supports defining and using requirements specifications at all stages of the product development cycle. Creating and using requirements specifications is in fact the role of all team members, even if the process is managed by a business analyst. A better quality of specifications leads to limiting the amount of necessary corrections in the project, reducing the total execution time and achieving better business results. The analyzed tool allows for the involvement of customers and other interested parties in the process of working on the requirements. It offers intuitive scenarios, process diagrams, use cases, and other visual and text techniques used to describe scenarios and reveal customer needs. The effective interaction and high transparency of the work of the team allows for quick agreement on the requirements among all interested parties [10]. The specificity of the tool made it possible to collect the functionalities which were particularly important in requirements development and communication with the client.

IBM Rational RequisitePro supports project teams in managing requirements, developing scenarios of use cases, tracking the origin of changes, the relationship with test cases, and also helps to reduce the number of necessary improvements and enhance the quality of the software. Managing complex projects is done through the use of detailed views of the origin presenting the relationships between elements. A big advantage of the analyzed tool is the possibility for efficient cooperation among geographically distributed teams, using a fully functional, scalable Web-based interface and a discussion organized in threads. Registering and analyzing information about requirements is possible due to detailed alignment and filtering attributes [9]. The tool characteristics indicate a particular emphasis on requirements management and a negligible effect on their development. The tool has provided a set of functionalities to manage requirements which were combined with the results of the Rational Requirements Composer tool analysis and thus the full area of requirements engineering was addressed.

3 The Concept of a Reference Model for Selecting Requirements Management Tools

Based on the discussed challenges associated with the selection of tools to support requirements engineering processes and best practices in the form of Rational products, a reference model was built which allows the identified problems to be minimized. The concept developed by the authors is to select functionalities and recommended integrations from the application of the model, necessary to support the needs of an organization in a specific area. The model includes functional and integrative best practices for the leading software engineering tools. The purpose of constructing the reference model based on the IBM Rational tools portfolio was to indicate the direction for the development of applications supporting requirements management and to apply the model in determining a strategy to improve departmental development.

The knowledge base constructed in accordance with this concept was supplemented with information about Open Source solutions. The authors carried out their task by functional and nonfunctional mapping onto the reference tool. The decomposition was achieved by a detailed analysis of the tools the authors, and direct contact with those responsible for the products (Product Manager) from IBM. The functional analysis of the selected tools revealed the limitations of the model at the very beginning of the study. Relying only on the functional decomposition of tools, it becomes impossible to assess their influence on an organization, in terms of implementation, thus overlooking a significant argument influencing both the effort and cost. The developed concept allows the tools' function to be separated from the established formal or agile process. On the one hand, the model can be applied to methodologies with a high level of formalism by choosing a larger pool of functionality, or on the other hand, it can head towards the Scrum and EclipseWay methodologies, by selecting only the required

elements. This separation from established processes allows the universal application of the model. It can be used in any organization regardless of the implemented process, with only expectations towards development in mind.

After analyzing best practices, the authors decided to apply the concept of architectural views which was already used earlier for modeling IT systems. The use of perspectives allows for easier management of a multi-dimensional structure, such as the reference model, while at the same time providing relevant information to the interested parties. The concept developed by P. Kruchten [11] introduces five perspectives (4+1 Architectural View Model), which affect different layers and aspects of the constructed system. These include functionality, business processes, logic, infrastructure and the internal structure of the solution. The model approach to modeling the architecture of IT systems allows the collaboration of many interested parties of a project on those elements which are most important to them, at the appropriate level of abstraction.

The authors are adapting this approach to embed it in the reference model for software life cycle management tools. The structure of this reference model includes the functionality of the tools, the infrastructure for implementation and integration, the life cycle disciplines with best practices, and the roles involved in the project. The model structure is dynamic and allows for the arbitrary selection of perspectives, so that a more or less detailed information model can be achieved.

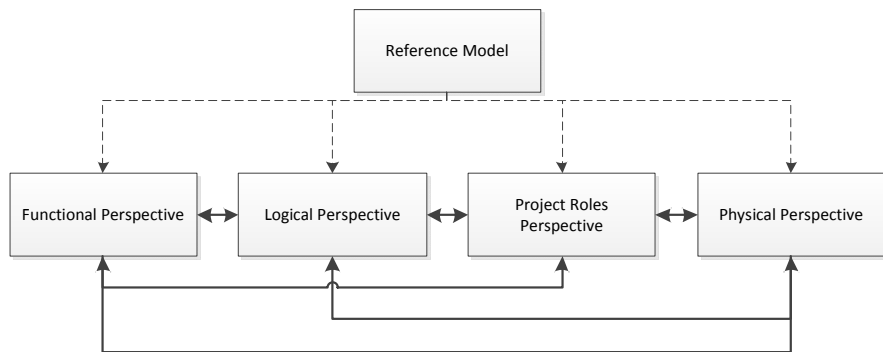


Fig. 2 The structure of a reference model for the selection of tools for the software development cycle

To specify the suggested perspectives in the reference model, elements of UML 2.4 were applied in the form of diagrams of use cases, implementation, packages and components. These diagrams fully support the concept of perspectives developed by P. Kruchten [11] and they allow the structural description of this model.

Functional Perspective – The basic architectural view allowing the analysis of functionalities associated with a particular branch of software engineering or a group of tools. Figure 3 presents a part of fully decomposed functional area for the branch of requirements engineering. The perspective shows the relationship

between the functionalities provided in the form of use cases and the relationships between them. The view was divided into disciplines in accordance with the Rational Unified Process methodology and the corresponding modifications relevant to the Agile concept. For the authors, the view constitutes a model of functional reference. The perspective is applicable when selecting the functionalities which the tools must have in the requirements management process and in the analysis of the relationship between them and other areas of their life cycle.

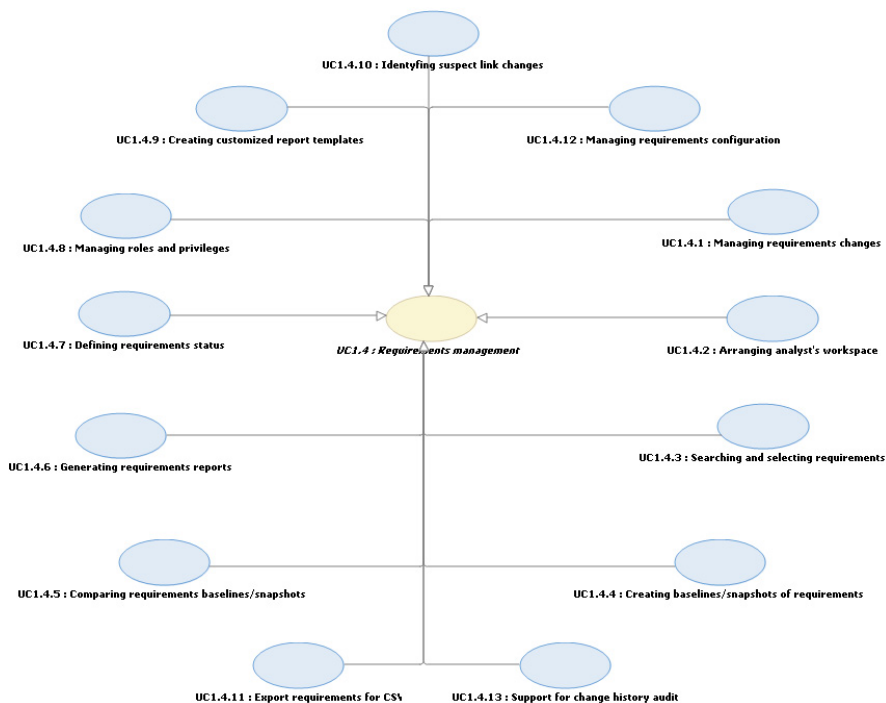


Fig. 3 The part of decomposition of the area for requirements management tools

Logical Perspective – The architectural view showing the relationship between the areas of manufacturing processes proposed in the functional perspective and the relationship between Open Source and IBM tools which support those relationships. An example of such mapping is shown in Figure 4, where functionality mapping of the functional perspective was carried out on the RaQuest tool in the logical view. On the presented visualization, the tool in question implements only a part of the whole standard set of functionalities of the reference model, which results from the limitations of the tool described in the specification.

For research purposes, an analogous functionality mapping was carried out on other leading Open Source tools. On the basis of chosen functionalities, the logical architecture allows the inference of what tools to choose and what areas of software engineering may be affected. It provides basic knowledge about tools and can be expanded at any time by a new portfolio. The view allows the user to quickly rule out a tool from further analysis, in terms of functionality, or, by extensive inquiry into the model, to analyze their use within the organization.

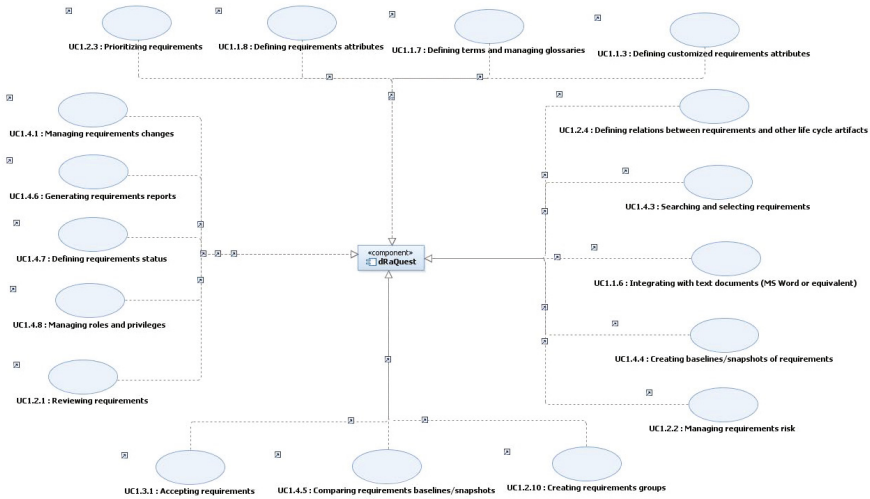


Fig. 4 An example of functionality mapping for Open Source (RaQuest) tools

Roles and Responsibilities Perspective – The architectural perspective presenting relationships between the project roles and the functionalities assigned to them in tools and their functions in projects. The view makes it possible to verify which interested parties should be involved in the implementation of chosen software and who will be its target user. It should be remembered that not every tool examines all areas of an issue at the same level of abstraction and can be dedicated to a variety of project roles. Looking at the developed reference model in a different way, we can look for a tool that will be able to support many project roles simultaneously, such as requirements management and testing processes.

Physical Perspective – This view presents integration or its possibility between Open Source tools and IBM Rational. The perspective shows the already existing built-in data flows and indicates how much work must be done to combine the tools, if they are not yet integrated. An example of such a connection might be integration between the requirements management tools and the architecture

management area. The view additionally describes sample deployment topologies, which can be used to estimate investments in the infrastructure related to the implementation of new tools. The physical perspective is the most technical in nature and allows a simple answer about whether it is a good long-term decision to change the development department strategy.

The architectural perspectives developed by the authors constitute the basis for further inference processes and further research. The views can at any time be completed with a new architectural set making the concept more detailed with new elements. The possibility of making modifications to the proposed approach allows for its adjustment to any expectations as well as the possibility of development with the use of groups of tools other than the IBM Rational and Open Source tools. A modular approach means that there is no need to go into the already existing perspectives, but complete the model incrementally with new elements or views. In developing the concept of the project, the authors anticipate the possibility of introducing new perspectives which will make the model more detailed in the area of additional non-functional requirements.

The concept of the further development of the model is based on open access to its resources for interested parties, who can simultaneously view but also complement the resources. Access to the presented perspectives, as well as the model itself, can be gained via a web-based interface and from the project environment. Both interfaces are shown in Figure 5, as access at browser-level is functionally limited, and is mainly used for browsing resources, and generating reports. In the IBM Rational Software Architect environment, used for tool development, we can supplement the perspectives with additional information, or create more complex queries.

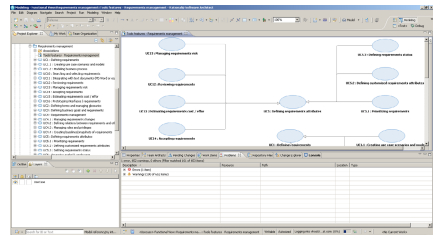


Fig. 5 An example of access to the referential model (WWW / modeling environment)

In this chapter, the authors have discussed the concept of a reference model based on architectural views, which provides a database of best practices related to technology. The proposed modular approach allows for the expansion of the solution with elements relevant to the target user or the integration of the model with reporting or inference systems. The authors foresee the possibility of extending the model with elements of expert systems allowing for analysis or additional inference through the use of open standards.

4 Possibility of Verifying the Developed Concept of the Reference Model

In this chapter the authors have discussed the possibility of verifying the concept of the reference model in terms of the chosen goal, namely the selection of Open Source tools for requirements management. The key reason for the implementation of the reference model was the ability to determine the further path of development for Open Source tools supporting the requirements engineering area. By using OCL language (Object Constraint Language) and the Birt report mechanism (Business Intelligence and Reporting Tools), the authors built a transformation mechanism where, based on a selected functionality of tools, we can compare solutions and adjust them to the needs of an organization. Reports supporting the tool selection process have been developed for the purpose of presenting the application of the reference model, as shown in Figure 6:

- functional comparison – the possibility of functionally compiling tools with one another and assessing their use for selected purposes,
- integration with other tools – the possibility of integrating the capabilities of the selected products with currently working solutions,
- implementation of designated functionalities – searching the reference model for tools possessing the selected functional solutions,
- tools dedicated to different project roles – the presentation of roles and functionalities dedicated to them within the chosen solution.

The verification of the model at this stage occurred by comparing the summary made by the model with the market statement obtained previously at the stage of project implementation. The authors have adopted the discussed method of verification due to the possibility of comparing previously obtained results with the real and documented choice of the target customer and their solutions analysis process. The presented approach was described as the closest to real attempts at the selection of technological solutions for supporting the requirements engineering process.

In analyzing the results, it could be observed that there was a convergence at the functional and integrative levels. The factors which have not been analyzed, but are important in the selection of tools, were costs, support procedures and further development, which may also play an important role in the selection of solutions. The authors assumed that in terms of the opportunities for development and the implementation of business needs, these are of secondary importance. This was justified by the inability to assess the gravity of such factors as the price, support procedures and further development in the case of Open Source tools and the lack of substantive links with technological support. It was assumed that after the analysis of the functional and nonfunctional requirements, and selecting a potential group of tools that meet the needs of the organization, it is possible to use additional parameters.

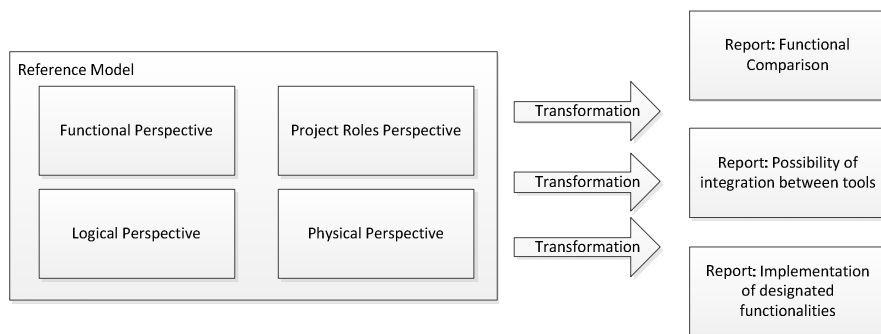


Fig. 6 The application of the model for the selection of tools for the software development cycle

Another way of verifying the reference model is in support for determining the tool application strategy in the organization. On the basis of examining customer maturity, the reference model and the currently available solutions, we can infer which tools should be used to fully address customer needs related to the process. The possibility of inference is based on the selection of functionalities required by the various processes, team size or other boundary parameters defined by the customer. The developed reference model stores functionalities and the relationship between the elements, namely the key elements from the perspective of inference. This will allow analysis of applied considerations to determine the best development path for the client’s portfolio and to compare it with the current assumptions. Figure 7 shows the application of the reference model in the

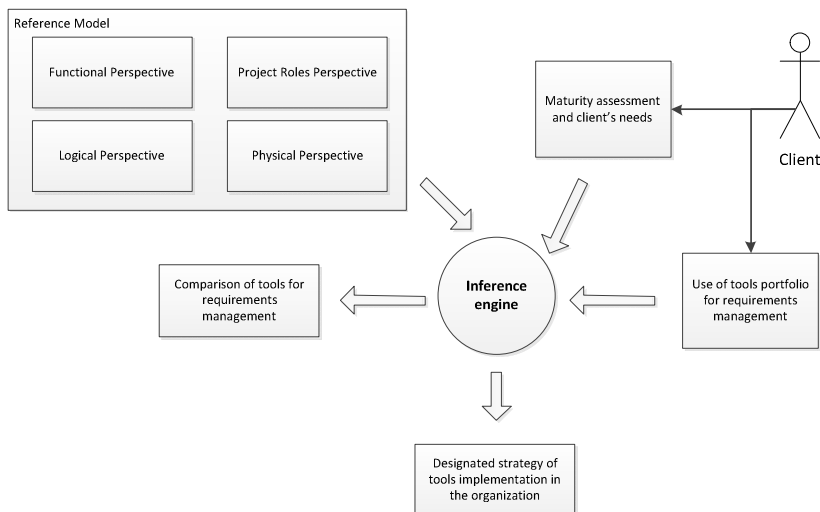


Fig. 7 Model for determining the development strategy

inference process of tool selection and the possibility of extending the concept by adding an inference engine. At present, the process of analysis and inference from the model is based on reporting mechanisms in the project environment. The expansion of the model with sets of rules which affect the choice of functionalities and an expert system that will significantly automate the inference and analysis processes is possible.

During research, the authors clearly identified the areas of application and benefits for the processes of requirements engineering and software development as the following:

- faster implementation of new tooling solutions,
- reduced time of implementing methodologies and processes,
- reduced risk of tool rejection,
- reduced costs for market analysis,
- better alignment with business needs,
- simulation possible before the actual implementation of technology.

These benefits were established on the basis of observations of the model's application in a practical way in companies providing business services in the telecommunications and pharmaceutical sectors, which were interested in improving their requirements management processes. The authors, by leading consulting projects, used the mechanisms built into the reference model for the visualization and simulation of architectural decisions in the context of tool selection.

The practical verification is consistent with those previously proposed methods of application which leads to the conclusion that these benefits will result from such use and development of the model. The validation of the reference model was achieved by its practical application in real advisory and implementation projects. The reference model was a supporting solution in compiling the client's needs with the real possibilities of Open Source tools available on the market. This chapter additionally shows the application of the model verified by the possibility of determining a long-term strategy for the tools applied in requirements management processes and complete manufacturing processes.

5 Assessment of the Developed Concept of the Reference Model

This article presents the concept of selecting and developing Open Source tools on the basis of a reference model for requirements management for software producing organizations. This paper has discussed the challenges related to tool selection in requirements management, as well as the branch of requirements engineering. After analyzing market-leading solutions in the field of management engineering, exemplary applications have been presented – ones which best meet

the requirements and customer needs. The authors assumed the IBM Rational Requirements Composer and IBM Rational RequisitePro tools to be model solutions, relying on market research and the independent assessment of some leading software supplying companies. The research process involved the functional and nonfunctional decomposition and the functional mapping of Open Source tools available on the market. The research resulted in a reference model describing the characteristics of the exemplary tools for the area of requirements management, at many different levels of abstraction dedicated for different recipients. The solution of the problem discussed earlier is modular which means that it can be freely adapted to the inference needs and as it is open, and it can be extended over time with new tools.

This publication presents the verification of the concept by applying the reference model for the selection and development of tools supporting requirements engineering processes on the basis of functionality and integration. On the basis of the model's application, the authors demonstrated the possibility of applying the model in practice, as well as its objective assessment.

The analysis conducted by the authors showed that the reference model should include all disciplines related to the software development cycle and present a comprehensive strategy for the construction of the portfolio of department development tools. The boundaries between the branches of the software development process are not clearly defined, making it impossible to describe just one specialization to the exclusion of others. The development of this reference model should be taken further by extending the number of disciplines and by integration between their components at different levels. An approach encompassing the full cycle of software development will allow for a complementary approach to the selection of tools in an organization and will enable the appointment of a coherent development strategy. Establishing a unified strategy in applying the reference model will reduce the amount of duplicated functionalities between tools and will provide better integration of solutions at the environment concept analysis stage.

The authors have indicated the possibility of the expansion of the reference model with additional architectural views, to allow thorough analysis and a better matching of the selected tools to the needs of organizations. The introduction of perspectives associated with software development methodologies or the assessment of the maturity process is an example of such an extension. The mapping of previously gathered functionalities on the methodologies will allow detailed matching of solutions to the current or target processes of customers.

The authors are planning their work and further research in these areas. The research project is a collaborative research project of the IBM Center of Advanced Studies and Gdańsk Technical University.

References

- [1] Ambler, S.: Survey Says: Agile Works in Practice - Agile software development methods and techniques are gaining traction. Addison-Wesley (2006)
- [2] Boehm, B.: A view of 20th and 21st century software engineering. In: Proceedings of 28th International Conference on Software Engineering (ICSE 2006), pp. 12–29. ACM Press (2006)
- [3] Broy, M.: Requirements engineering as a key to holistic software quality. In: Levi, A., Savaş, E., Yenigün, H., Balcısoy, S., Saygın, Y. (eds.) ISCIS 2006. LNCS, vol. 4263, pp. 24–34. Springer, Heidelberg (2006)
- [4] Chrabski, B., Zawistowski, P.: Architecture of information systems in modern telecommunication companies. *Przegląd Telekomunikacyjny* 12/2011, Bogdan Zbierchowski, Krystyn Plewko, Hanna Wasiak, SIGMA NOT, Warszawa, pp. s.1626–s.1631 (2011) ISSN 1230-3496
- [5] Why do projects fail?, reasons of IT projects failures (2009), <http://workflow.com.pl/pl/tag/the-standish-group>
- [6] Global Association for Software Quality Requirements Engineering Qualifications Board, Syllabus dla Certyfikacji ReqB (2011)
- [7] Hooks, I.F., Farry Kristian, A.: Customer-Centered Products: Creating Successful Products Through Smart Requirements Management. Amacom, New Nork (2001)
- [8] IBM, Jazz Plan Jam, Rational Innovate Conference (2012), <https://jazz.ideajam.net>
- [9] IBM, IBM Rational RequisitePro Datasheet (2007), <ftp://ftp.software.ibm.com/software/rational/web/datasheets/reqpro.pdf>
- [10] IBM, IBM Rational Requirements Composer Datasheet (2011), <http://public.dhe.ibm.com/common/ssi/ecm/en/rad14076usen/RAD14076USEN.PDF>
- [11] Kruchten, P.: Architectural Blueprints—The “4+1” View Model of Software Achitecture. *IEEE Software* 12(6) (1995)
- [12] Nikula, U., Sajaniemi, J., Kälviäinen, H.: A State-of-the-practice Survey on Requirements Engineering in Small- and Medium-sized Enterprises, Research Report 1, Telecom Business Research Center Lappeenranta (2000)
- [13] Nuseibeh, B., Easterbrook, S.: Requirements engineering: a roadmap. In: Proceedings of the Conference on the Future of Software Engineering, pp. 35–46. ACM Press (2000)
- [14] Ovum, Ovum Application Lifecycle Management Report (2009), <http://www.rationalindia.in/ibm-rational-the-recognized-leader-in-alm/>
- [15] Schmidt, P.: Reasons for failure of projects, why it is worth using a methodology (2008), http://pmanager.pl/index.php?option=com_content&view=article&id=145&Itemid=53
- [16] Wiegers, K.E.: More about Software Requirements. Microsoft Press (2006)

A Probabilistic Approach to the Count-To-Infinity Problem in Distance-Vector Routing Algorithms

Adam Czubak

Institute of Mathematics and Informatics, Opole University
ul. Oleska 48, 45-052 Opole, Poland
adam.czubak@math.uni.opole.pl

Abstract. Count-to-infinity problem is characteristic for routing algorithms based on the distributed implementation of the classical Bellman-Ford algorithm. In this paper a probabilistic solution to this problem is proposed. It is argued that by the use of a Bloom Filter added to the routing message the routing loops will with high probability not form. An experimental analysis of this solution for use in Wireless Sensor Networks in practice is also included.

1 Introduction

The classical Bellman-Ford algorithm is used for computing shortest paths between any two given nodes in a graph. The distributed version of this algorithm (DBF) was the origin of most of the distance-vector routing algorithms and protocols used today. Its advantages are:

- Low communicational complexity;
- Low computational complexity;
- Low space complexity;
- Ease of troubleshooting in case of an issue.

But all algorithms derived from DBF, like their predecessor, suffer from a count-to-infinity problem. It causes routing loops to form in the network topology in case of a link failure. The solutions to this problem used in practice involve additional administrative configuration of the routing protocol (RIPv2 and RIPv6 protocols) or sending a list of all the visited vertices along the path within the routing update (BGP and MP-BGP protocols).

2 Preliminaries

2.1 Count-To-Infinity Problem

The count-to-infinity problem is characteristic for a certain group of distributed algorithms. The advantage of utilizing only a limited scope of information

regarding the network is very tempting on one hand, but on the other it bears a certain risk. Since the nodes are not aware of the whole network structure, in case of a change in the topology the nodes do not have up-to-date information at hand. At least for a certain amount of time. The information is globally updated over time, in some locations sooner, in other later. In certain cases this divergence of the knowledge about the network may lead to creation of routing loops. Let's consider a simple example depicted in Figure 1. We represent the network as a graph $G = (V, E)$, where $V = \{1, 2, 3, 4, 5\}$, $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}\}$ (Fig.1).

Since none of the nodes is aware of the network topology and all is know are local neighborhood subgraphs of G , a situation may occur, where a node v_2 adjacent to v_1 informs v_1 that there is a path to destination available that was actually received originally from v_1 but is no longer valid. Such a situation, called count-to-infinity problem is shown in Figure 1. Let's analyze this problem more thoroughly:

1. In the first phase every node after at most 4 time units has a path to the destination node v_5 and the network is therefore converged.
2. In phase two the link between nodes v_4 and v_5 malfunctions. Node v_4 loses its path to v_5 but after a short while, it receives a new path from v_3 . This behavior is erroneous, because the path through v_3 leads through v_4 , which is now unavailable, but since only a limited, local scope of the network is known to nodes, there is no way of knowing that the new path is actually a loop.
3. In the third phase the node v_4 itself informs its neighboring nodes that it has a route to v_5 . Node v_3 updates its best distance to v_5
4. In phase four the node v_3 sends its update to neighbors v_2 and v_4 ;
5. Consequently the route update traverses the nodes in a loop between v_4 and v_3 .

There are several known solutions to the count-to-infinity problem:

- Every routing update sent may be equipped with a list of traversed nodes. Every node the update visits would add its ID to the list. This method is used nowadays in WAN networks in BGP-4 (Border Gateway Protocol) and MP-BGP (Multiprotocol BGP Extensions) protocols. The solution is not feasible for large and dense networks with huge number of nodes like WSNs;
- Split Horizon is rule, which states, that an update may never be sent to its source node. This eliminates only two-node loops and is not suited for multi-point and wireless networks;
- A limit on the path length may be imposed;
- A hold-down timer may force a node to ignore an update for a certain amount of time, to stop the spreading of false information;
- Route poisoning marks a specific route actively as unavailable;
- Sequential numbers for routing updates were proposed in some networks[7,8].

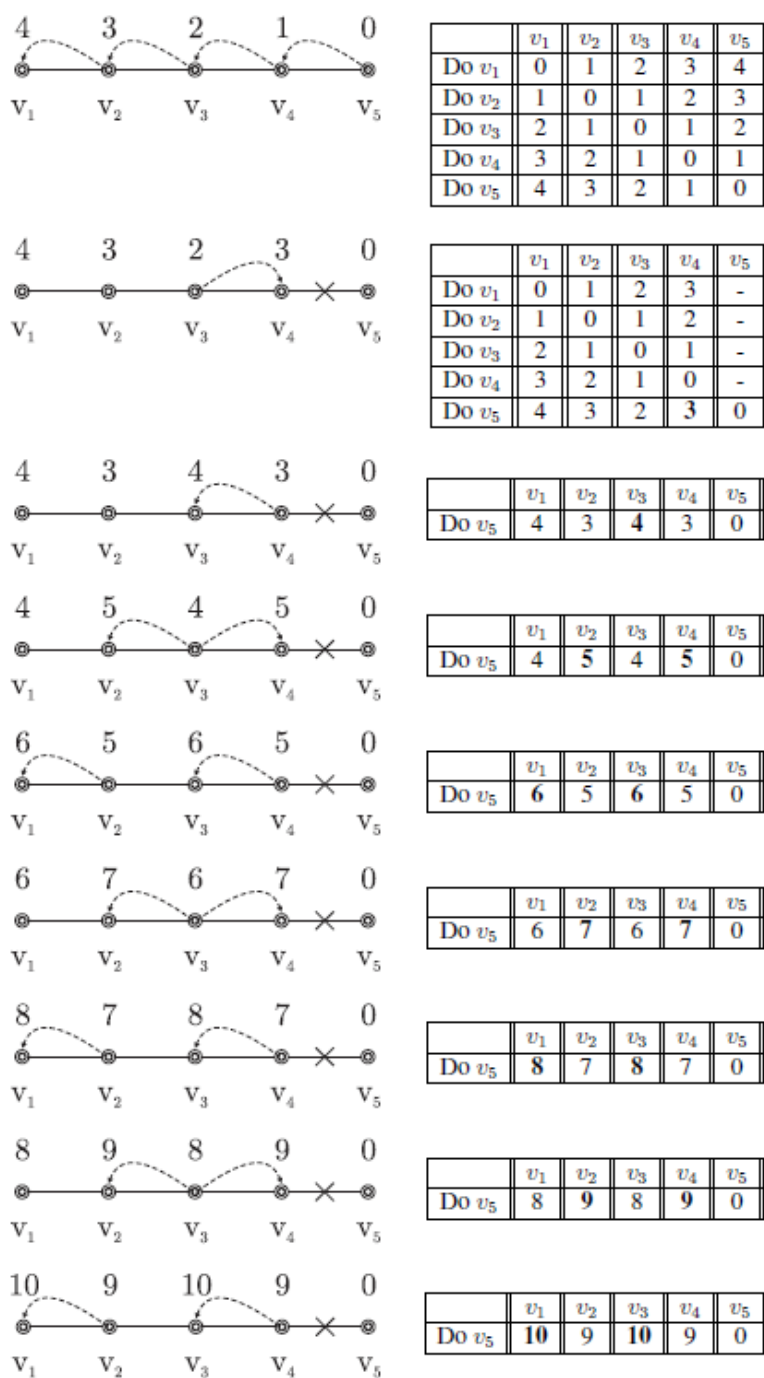


Fig. 1 Visualization of the count-to-infinity problem

2.2 Bloom Filters

A Bloom filter is a data structure for the approximate set membership problem [1], it may represent a set of elements. It consists of an array of m bits $BF[0]$ to $BF[m-1]$ initially all set to 0. A Bloom Filter uses k independent hashing functions $\{h_1, \dots, h_k\}$ with range $\{0, \dots, m-1\}$.

Bloom filter work as follows. Let's assume there is a space Q and a set of elements $S \subseteq Q$, such that $S = \{s_1, s_2, \dots, s_{|S|}\}$. The target here is to project Q onto a much smaller BF, so that it would be possible to give an answer to a question whether some $x \in S$. As a projecting function we will utilize k -hashing functions $\{h_i(y) : i=1 \dots k\}$, $h_i : Q \rightarrow BF$. We have an m -bit array at our disposal (Table 1).

Table 1 An empty Bloom filter

0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---

Now we execute k -hashing functions on every element of the set S . If $h_i(s_j) = a$, so if the hashing function returns a number, set a bit $BF[a] = 1$. It is possible, that this certain bit will be set multiple times, but only the first change has an effect (Table 2):

Table 2 A Bloom filter containing elements

0	0	1	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---

In order to answer a question, whether some element $x \in S$ was placed in the BF we need to check whether under the indices returned by the k -hashing functions $\{h_i(x) : i = 1 \dots k\}$ in the array BF the corresponding bits are set to 1, i.e. whether

$$\bigvee_{i=1 \dots k} (BF[h_i(x)] = \text{true})$$

With some probability it may happen, that an element which seems to be contained within the filter is actually not contained, though the hashing functions return bits that are set to 1 in the filter. Previously inserted elements must have set those bits earlier. There are a couple of parameters, we can tune to affect the behavior of a Bloom filter and improve the probability in our favor:

- The size of the BF array;
- The amount k of the hashing functions;
- The amount of the elements inserted into BF.

It is worth mentioning, that the size of the filter does not depend on the amount of the elements in the set. The cost one must pay for a constant space complexity is the fact, that the filter is probabilistic in nature and to check whether a certain element is within the filter, it may, with some probability, return an incorrect answer, called *false-positive* or a *collision*. Burton Bloom in his paper [1] placed

messages inside of a filter and later on verified whether such a message was received before or not.

The main goal here is to minimize the probability of occurrence of a false positive. Let's notice, that after insertion of n elements, the probability that a certain bit is not set equals:

$$\Pr(\text{BF}[a] = \text{false}) = (1 - 1/m)^{kn}$$

So the probability of occurrence of a false positive is precisely and asymptotically [6]:

$$\Pr(\text{False Positive}) = (1 - (1 - 1/m)^{kn})^k \approx (1 - e^{-kn/m})^k$$

Now let's assume, that m and n are known, let's optimize the number of required hashing functions. The derivative of the function $g = k \ln(1 - e^{-kn/m})$ after k and the zero of the function points to a global minimum $k = (\ln 2)(n/m)$. So the probability of occurrence of the false positive equals to:

$$\Pr(\text{False Positive}) = (1/2)^k \approx (0.6185)^{n/m}$$

From the above follows, that the size of the Bloom filter in bits must be a couple of times larger than the amount of elements that we intend to insert. A well engineered Bloom filter should return incorrect answers with negligible probability. The capabilities of Bloom filters are as follows:

- Inserting an element;
- Answering membership queries;
 - If negative, it is certain that the element was never inserted;
 - If positive it may be a false positive;
- The sum of two BF is achieved by a simple bitwise AND operation

In order to minimize the probability of a false positive we can:

- Increase the size of the BF array;
- Manipulate the amount of used hashing functions.

The advantages of Bloom filters are:

- Space efficiency, $\text{Space}(\text{BF}) \ll \text{Space}(S)$;
- The computational complexity of answering membership queries does not depend on the size of $|S|$ or $|Q|$. It depends only on the time required to perform the hashing and additionally: $\text{BF} \wedge \text{hash}$. So it equals to $O(1)$;
- The BF will never return a false negative answer.

The disadvantages of Bloom filters are:

- With some probability a BF may return a false positive;
- It is not possible to remove an element. It would require to unset bits in the BF, but those might have been set previously so resetting might compromise the structure and result in false negatives. The solution here is to use counting Bloom filters [34], where instead of bits natural numbers are used. But this increases the size of BF.

Some applications of BF are [10]:

- In Squid proxy server;
- In P2P networks, where the server stores a list of objects (i.e. files) available in a BF

3 Bloom Filters and the Count-To-Infinity Problem

As mentioned earlier, one of the methods to tackle the count-to-infinity problem is to append a list of previously traversed nodes to the routing update. In this simple way every node can check whether a loop was created or not by checking if it is on the list. We propose to append a fixed-size Bloom filter instead of an expanding list to the routing update. The BF would contain the previously visited nodes and every node would be capable to check with some probability whether a loop was formed or not.

For this proposed solution a testbed was chosen. A recently announced distributed algorithm called Lifespan-Aware Routing (LAR) [4] was developed for use specifically in Wireless Sensor Networks [3]. It is a distance-vector routing algorithms based on distributed Bellman-Ford algorithm and suffers from the count-to-infinity problem.

Before implementing BF for use in WSN certain questions must be answered:

1. Do sensors have the required resources to use Bloom filters? It turns out that the computing power and hardware are more than enough.
2. Can sensors generate hashes? Both industry standard RC5 and CBC-MAC are implemented in TinyOS, an operating system used in WSNs.
3. How many hashing functions should be used and what should be the size of a BF? In WSNs the Q space corresponds to node identifiers, which are unique. In the available literature WSNs consist of no more than thousands of nodes.
4. How large may the BF be regarding communication capabilities of WSNs? In WSNs the IEEE802.15.4 standard is widely used. The maximum size of the transferred data after subtracting the header payload is between 122 and 102 bytes, depending on the type of addresses used (Fig. 2)

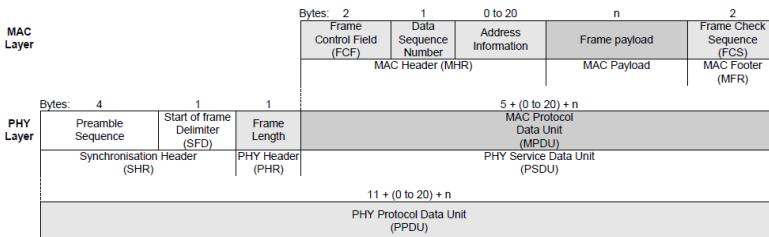


Fig. 2 IEEE 802.15.4 Frame Format

Now let's consider a random wireless sensor network. Lifespan-Aware Routing algorithm forms a spanning-tree in a distributed manner. It seems that it is correct to presume, that the collector (common destination node) is in the center of the network. This presumption is very appealing because it implies that the BF has to store the identifiers of the nodes in the size of the radius path. But in real-life scenarios it turns out, that the collecting node is localized on the edge of the network (Fig. 3) and the BF should be large enough to store the diameter of the network.

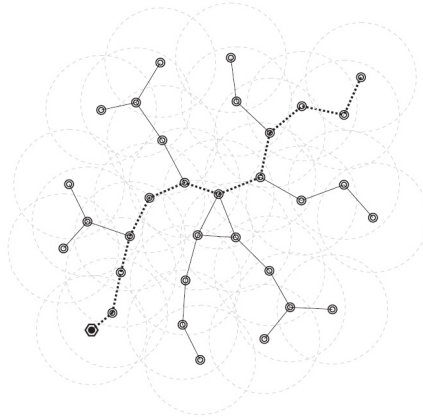


Fig. 3 Diameter of a random WSN topology

So the question arises regarding the diameter size of a typical WSN. In the year 2011 Chung, Horn i Lu [2] researched the subject of random spanning trees and their diameters. The results are particularly important, since these did not focus on minimal spanning trees and LAR forms a spanning tree which is not necessarily minimal. A spanning tree is an acyclic subgraph of graph G containing $|V| - 1$ edges. If T is a random spanning tree of graph G , then the size of diameter $diam(T)$ with high probability is between [2]:

$$c_1 \sqrt{|V|} \leq diam(T) \leq c_2 \sqrt{|V| \log |V|}$$

The constants c_1 and c_2 before the lower and upper bounds depend on the characteristics of a graph G like: average node degree, minimal node degree, second order node degree, and spectral gap of a graph. The above result although important does not give a tangible answer to our question on diameter length of WSNs.

From [5, 9] we can deduce a more fixed estimation. The authors deal with Bluetooth networks which bear a lot of similarities to WSNs regarding communication capabilities. According to their work it is safe to assume, that a diameter of a random graph equals $\sqrt{|V|}$. Form available literature we can assume that a WSN consists of no more 10.000 nodes. It means, that that we need to store 100 nodes in the Bloom filter.

Table 3 Correlation between the amount of hash functions and the size of the Bloom filter

$ FB $	$ V $	1	2	3	4	5	6	7
957	100	0,099268	0,035601	0,019513	0,013643	0,011182	0,010235	0,01014
958	100	0,09917	0,035534	0,019461	0,013597	0,011138	0,010189	0,010089
959	100	0,099072	0,035468	0,019409	0,013551	0,011094	0,010143	0,01004
960	100	0,098974	0,035401	0,019358	0,013506	0,011105	0,010098	0,00999
961	100	0,098876	0,035335	0,019306	0,013461	0,011006	0,010053	0,009941
962	100	0,098778	0,035269	0,019255	0,013415	0,010963	0,010007	0,009891
963	100	0,098681	0,035203	0,019204	0,013371	0,010919	0,009963	0,009843

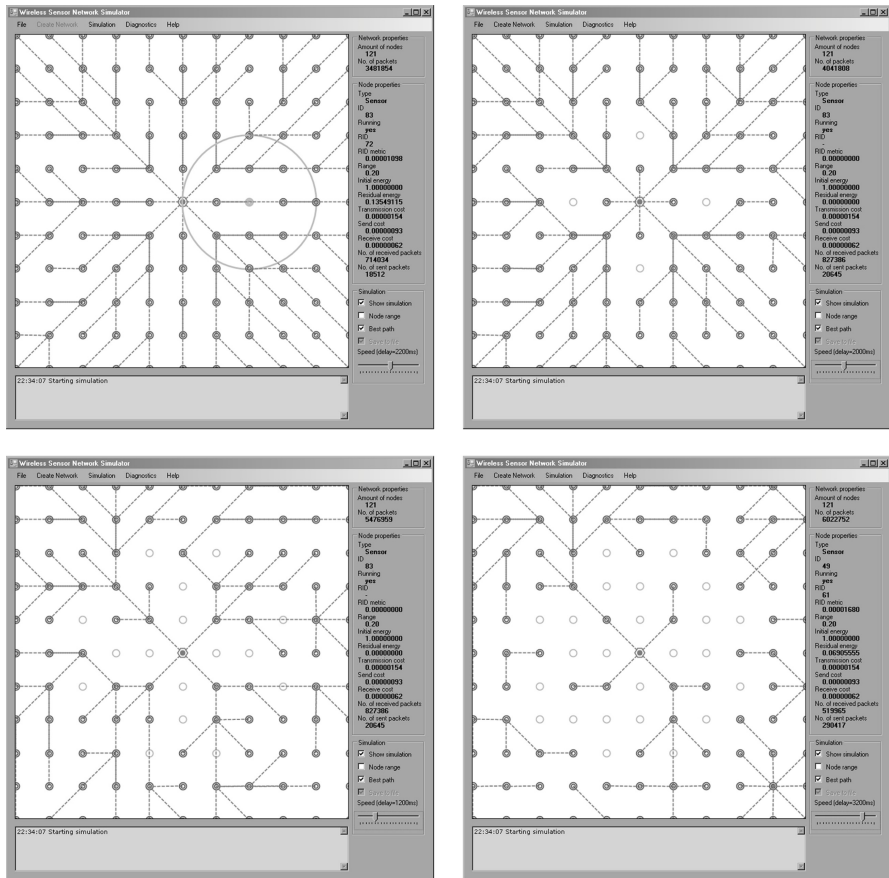


Fig. 4 WSN simulation with Bloom filter applied

In order to achieve the probability of 0.943% that a false positive will occur for 100 identifiers stored in the BF we need to use 5 hashing functions and a BF array ten times the number of elements to store. So it is sufficient to send additional 1000 bits in a routing update to achieve a 99.043% loop-free topology.

From the size of the IEEE 802.15.4 frame it would seem that the BF fits inside a single frame, but it is actually 125 bytes long and the frame carries a maximum 122 bytes of data. Additional analysis of the correlation between the number of hash functions and BF size shows, that by using 7 hash functions the size of BF may be decreased to 960 bits while still achieving 0.999% probability that a false positive occurs (Table 3). 960 bit fit inside a single 802.15.4 frame, so it is sufficient to send a single frame more to achieve a 99% loop-free topology in WSNs.

The Lifespan-Aware Routing algorithm was implemented in a custom made WSN simulator with the Bloom filters added to the network updates. Random topologies of the size 11×11 were conducted 100 times, no occurrence whatsoever of the count-to-infinity was discovered (Fig. 4)

4 Summary

The count-to-infinity problem was studied in this paper. A probabilistic solution was proposed. The main idea of the solution is to add to the routing update of a distributed distance-vector algorithm a Bloom filter containing all the nodes which the update traversed. The solution was evaluated for use in Wireless Sensor Networks and implemented in Lifespan-Aware Routing algorithm. Additional research into the frame standard and Bloom filter size allowed for making the statement, that an additional frame per routing update secures a 99% loop-free network topology in the case of the IEEE802.15.4 communication standard. The conducted simulation experiments show the feasibility of this solution in practice use in Wireless Sensor Networks.

References

- [1] Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM* 13, 422–426 (1970)
- [2] Chung, F., Horn, P., Lu, L.: Diameter of random spanning trees in a given graph. *Journal of Graph Theory* 69(3) (2012)
- [3] Czubak, A., Wojtanowski, J.: On Applications of Wireless Sensor Networks. In: Tkacz, E., Kapczynski, A. (eds.) *Internet – Technical Development and Applications*. AISC, vol. 64, pp. 91–99. Springer, Heidelberg (2009)
- [4] Czubak, A., Wojtanowski, J.: Lifespan-Aware Routing for Wireless Sensor Networks. In: Jędrzejowicz, P., Nguyen, N.T., Howlet, R.J., Jain, L.C. (eds.) *KES-AMSTA 2010, Part II*. LNCS, vol. 6071, pp. 72–81. Springer, Heidelberg (2010)
- [5] Ellis, R.B., Jia, X., Yan, C.: On random points in the unit disk. *Random Struct. Algorithms* 29(1), 14–25 (2006)

- [6] Mitzenmacher, M., Upfal, E.: *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York (2005)
- [7] Perkins, C.E., Bhagwat, P.: Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers. In: *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, pp. 234–244. ACM, New York (1994), doi:10.1145/190314.190336
- [8] Perkins, C.E., Royer, E.M.: Ad-hoc On-Demand Distance Vector Routing. In: *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100 (1999), doi:10.1109/MCSA.1999.749281
- [9] Pettarin, A.: *On the diameter of Bluetooth-based adhoc networks*. Ph.D. Thesis, University of Padua, Italy (2008)
- [10] Ramakrishna, M.V.: Practical performance of Bloom filters and parallel free-text searching. *Communications of the ACM* 32, 1237–1239 (1989)

A Quality Estimation of Mutation Clustering in C# Programs

Anna Derezińska

Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
A.Derezinska@ii.pw.edu.pl

Abstract. Mutation testing tasks are expensive in time and resources. Different cost reduction methods were developed to cope with this problem. In this chapter experimental evaluation of mutation clustering is presented. The approach was applied for object-oriented and standard mutation testing of C# programs. The quality metric was used to compare different solutions. It calculates a tradeoff between mutations score accuracy and mutation costs in terms of number of mutants and number of tests. The results show a substantive decrease in number of mutants and tests while suffering a small decline of mutation score accuracy. However the outcome is not superior to other cost reduction methods, as selective mutation or mutant sampling.

1 Introduction

In mutation testing many faulty versions (so-called *mutants*) of a program under test are generated by application of mutation operators. If any test from a given test set detects an abnormal behavior of a mutant, the mutant is set to be *killed*. The ability of a test set to reveal faults specified by mutation operators is named *mutation score (MS)* and measured as a ratio of the number of killed mutants over the number of all non-equivalent mutants. An *equivalent* mutant generates the same outcomes as the original program and cannot be killed by any test. Mutation operators define various kinds of faults. Standard operators deal with expressions and structural features common to all general purpose programming languages, whereas object-oriented (OO) operators with flaws specific to OO languages [1].

An important obstacle of mutation testing approach is the computational expense. For one program many mutants can be generated and one mutant is run with many test cases. Several cost reduction techniques to mutation testing were proposed based on “do smarter”, “do faster” or “do fewer” approaches [1,2]. The mutation clustering belongs to a “do fewer” method that tries to execute fewer mutants against fewer test cases.

In mutation clustering a set of mutants is divided into disjoint subsets, so-called groups, based on the ability of tests to kill these mutants. Various groups of mutants can be killed by the same, or similar, subsets of tests. In the further testing

process, only one mutant representing a group is applied instead of all mutants from the group. Division of mutants can be realized using one of the clustering algorithms, such as agglomerative hierarchical or K-means clustering [3, 4] or by static domain analysis [5].

The analysis on mutation clustering was performed on C programs using standard mutation operators [3], and on an exemplary, simple Java program [5]. It was not shown how the approach will scale up for bigger-size and practically used programs. Another open question was whether the clustering method can give benefits to the object-oriented mutation testing, as according to the author's experience [6,7] the object-oriented mutation evaluates with different characteristics than mutation with standard operators. Mutation testing of C# programs is supported by the CREAM tool [6,8,9]. In order to effectively perform experiments on various cost reduction methods, an extension to CREAM was implemented [7,10].

The aim of the research presented in this chapter is examination whether the clustering method is worthwhile in mutation testing of C# programs, both in terms of standard and object-oriented operators. The tradeoff between the quality of mutation testing result (*MS*) and its cost (number of mutants and tests) is quantitatively evaluated with assessment of an original, tool supported metric [7].

The remainder of this chapter is organized as follows. Next Section describes related work. In Section 3 the main methodological issues are discussed. The experimental set-up is presented in Section 4. Section 5 gives experiment results and their analysis. Finally, Section 6 concludes the work.

2 Related Work

Promising results of mutation clustering with standard operators and C programs were presented in [3]. They showed, for example, that using a substantially reduced number of mutants (13%) and of tests (8%) we can obtain almost the same mutation score, i.e. 99%. In experiments with C# programs so good results were obtained neither with standard nor with OO operators.

Experiments in [3] focus on the assessment of potential clustering benefits, similarly as in this chapter. Therefore clustering was based on results of runs of all mutants against all test cases. Ji at al. proposed a practical approach in which clustering is based on static domain analysis [5]. The experiment proved that the method is applicable and dealt with a small Java program giving the encouraging results (e.g. 25% mutants with 62% tests gave 94% of mutation score).

The mostly studied "do fewer" method was selective mutation [11-14], in which only subset of mutation operators is used. Five standard operators were recognized as selective in experiments with Fortran programs [11]. The research on OO mutation was neither so promising nor so conclusive [7,14]. A method of mutant sampling, based on random selection of mutants, gave good results (10% of mutants with 16% loss of *MS* accuracy) for standard mutation in Fortran [15]. Sampling according to different criteria was studied in [10] for C# programs.

There are several tools for mutation testing of Java programs, but the only tools that support mutation testing of C# programs with some standard and OO operators are those implemented under the author's supervision: CREAM [6,8,9] and the prototype ILMutator [16]. The latter injects faults directly into Intermediate Language of .NET and therefore speeds up mutant generation. However the cost of test execution remains the same as using CREAM.

3 Methodology

In experiments discussed in this chapter the agglomerative clustering algorithm was applied [4]. Below its general idea is presented. Next, the experiment scenario and quality metric are discussed.

3.1 Clustering Algorithm

A clustering algorithm returns the division of mutants for a given set of mutants M and a set of test T that kill the mutants. The algorithm is characterized by a threshold parameter K reflecting a similarity of mutant groups. Two groups are similar with K degree, if the number of tests that kill at least one mutant from one group and kill none mutant from the second group is equal to K .

The general idea of the algorithm can be described in following steps:

- 1) First, for each mutant from the set M , a mutant group is initialized. Therefore there are $|M|$ groups of one element, where $|X|$ denotes cardinality of set X .
- 2) A temporary group similarity value is set to 0 ($i = 0$).
- 3) All pairs of current groups are compared. Two groups are merged if the similarity of the groups is less than the temporary group similarity value (i). If there are no more pairs of groups to be merged we go to the next step.
- 4) The algorithm stops if the current group similarity value reaches the algorithm threshold ($i = K$). Otherwise, the temporary similarity value is incremented ($i++$) and the algorithm is continued with the step 3.

For example, given a set of mutants $M = \{m1, m2, m3, m4\}$ killed by the test sets $\{t1, t2\}$, $\{t1, t2\}$, $\{t1\}$, $\{t2, t3\}$ accordingly, and the parameter $K=1$, we obtain the following two groups of mutants $\{m1, m2, m3\}$ and $\{m4\}$. The first group is killed by the test set $\{t1, t2\}$, whereas the second group is killed by tests $\{t2, t3\}$.

3.2 Experimental Scenario on Mutation Clustering

In experiments on cost reduction methods we answer a question how a reduced number of mutants is able to assess the test quality (MS) in comparison to all mutants that could be generated. Moreover we look for minimal test sets that could be as effective in revealing faults as the reference test set. The experiments on mutation clustering were designed according to the following scenario:

- A) Using a given set of mutation operators, all first order mutants of a program under test are generated (mutant set M_{All}).
- B) All mutants from M_{All} are run against all tests from a considered set T_{All} . The resulting mutant execution matrix states for each pair \langle mutant m , test t \rangle whether the mutant m is killed by the test t or not.
- C) A parameter K of the clustering algorithm is selected. Disjoint mutant groups are determined by the clustering algorithm for given mutants M_{All} , test set T_{All} and parameter K .
- C1) A subset of mutants $M_{C1} \subseteq M_{All}$ is created by selection of one representative mutant from each mutant group. Mutation score $MS_{C1max} = MS(M_{C1}, T_{All})$ is calculated assuming that mutants from this subset were tested by all tests.
- C2) In order to optimize a test set, a collection L of test subsets of T_{All} is created. Any test set in L has a minimal number of tests and gives the mutation score equal to MS_{C1max} (from step C1). Tests sets meeting those requirements can be generated using prime implicant of a monotonous Boolean function [17]. The collection L includes either all test sets of this kind, or a limited number $TestSetLimit$ of such sets. The value of $TestSetLimit$ is a parameter of an experiment.
- C3) For any test set included in L a mutation score is calculated as if all mutants from M_{All} were tested by the test set.
- C4) The average mutation score MS_{avg} is calculated from the results of step C3.

Investigating an impact of the clustering threshold on the mutation results, we can repeat steps C(C1-C4) for different values of K . Next the final statistics and quality metrics are calculated.

3.3 Quality Metric

The primary metric used for evaluating results on mutation testing process is the mutation score (MS). The original mutation score $MS_{orig} = MS(M_{All}, T_{All})$ is calculated using execution results of all mutants from set M_{All} and all tests from set T_{All} . If a reduced number of mutants ($M_i \subseteq M_{All}$) and/or a reduced number of tests ($T_i \subseteq T_{All}$) are taken into account, the mutation score can be less accurate than the original one. In order to estimate the mutation testing approach not only in terms of the mutation score accuracy but also the cost factors, the quality metrics were proposed [7]. Using the metrics it is possible to compare results of different programs and different experiments, as it is based on a normalization function and takes therefore values from 0 to 1.

The quality metric EQ applied in the analysis of mutant clustering is a weighted sum of three components (Eq. 1)

$$EQ(W_{MS}, W_T, W_M) = I(W_{MS} * I(S_{MS}) + W_T * I(Z_T) + W_M * I(Z_M)) \quad (1)$$

The metric is based on three dependent variables that assess a decline of mutation score accuracy (S_{MS}), a reduced number of tests required to kill mutants (Z_T), and a

reduced number of mutants (Z_M). Contributions of these factors to the metric are calibrated by weight coefficients W_{MS} , W_T , W_M , which sum must be equal to 1. $I()$ denotes a normalization function. The normalization is performed for all results in an experiment. Detailed formulae of the variable computation are given in [7], where the quality metric was applied for quality evaluation of selective mutation.

4 Experimental Set-Up

The mutation testing process discussed in this chapter dealt with *first order mutation* - a mutant is created by introducing one fault specified by one mutation operator in a program under test, and *strong mutation* - a mutant is recognized to be killed if a result of at least one test differs from the result of the original program.

Three widely used, open-source programs related to different domains and various authors were used in the experimental study. The basic statistics of the programs are given in Table 1. The tests associated with the first project - Enterprise Logging were unit tests designed and run with MSTest, a part of the Microsoft Visual Studio, whereas tests of Castle and Mono-Gendarme were NUnit tests.

Table 1 Subject programs and their statistics

No	Program	LOC		Classes & Interfaces	
		with tests	without tests	with tests	without tests
1	Enterprise Logging http://entlib.codeplex.com	87552	57885	991	587
2	Castle http://www.castleproject.org	54496	41288	724	493
3	Mono-Gendarme http://www.mono-project.com/Gendarme	51228	25692	907	171
Sum		193276	124865	2622	1251

The experiments were carried out with the CREAM (CREATOR of Mutants) tool a mutation system for C# programs mutated at the syntax tree level [6,8,9]. It is the most mature mutation system of C# applications. The latest version of the tool was extended with a wizard in order to efficiently perform experimental study on cost reduction techniques. The extension assists in creating mutants, executing tests, and evaluating test results in respect to three methods: mutation operator selection, mutant sampling and mutation clustering.

The experimental scenario from Sec. 3.2 and the whole analysis were performed independently for two sets of mutation operators: 18 object-oriented and 8 standard ones implemented in CREAM v3 [7]. The standard operators cover the five operators distinguished to be selective [11].

5 Experiment Results and Quality Analysis

The basic mutation testing results of subject programs are summarized in Table 2.

The first row includes numbers of mutants referring only to the covered code. The programs were covered by their unit tests in 82%, 77% and 87% respectively. None of the mutants generated for uncovered code were killed. Therefore in the calculations of the mutation score only the covered code was taken into account.

For each program, mutants were run against all tests T_{All} associated with the program. The numbers of killed mutants are given in the second row.

Some generated mutants can be equivalent. Equivalent mutants were manually detected by analyzing mutants generated by selected operators (with the highest number of not killed mutants and those easily to be analyzed). The numbers of recognized equivalent mutants are listed in the third row. However, some equivalent mutants could remain undetected. Covered and not recognized as equivalent mutants were counted as a set of all mutants M_{All} generated by either OO or standard operators, respectively. Basing of this data the original mutation score MS_{orig} was calculated. It is given in the last row and will be counted as a reference value.

Table 2 Mutation results - number of mutants generated, killed, equivalent and mutation score

	1. Enterprise Logging		2.Castle		3. Mono-Gendarme	
	O-O	Standard	O-O	Standard	O-O	Standard
Generated covered mutants	1341	1683	1208	2379	998	4153
Killed mutants	558	1151	701	1611	478	3009
Equivalent mutants	438	60	143	60	143	79
Mutation Score (MS_{orig}) [%]	61,79%	70,92%	65,82%	69,56%	55,91%	73,86%

The test results of all mutants were used in further steps C(C1-C4) of the mutation clustering scenario (Sec. 3.2) performed under the following assumptions:

- the clustering parameter K varied from 0 to 19,
- *TestSetLimit* - the number of minimal test sets in collection L was set to 15.

Average mutation score (step C4) calculated for different values of the parameter $K=1..19$ is shown in Table 3. This value reflects an average mutation result that could be obtained if we used not all mutants but only its subset - representatives of groups determined with a given K parameter. In general, higher values of K result in the drop of mutation score, although the functions are not strictly monotonous. This effect is caused by selection of one mutant representing a group.

If no clustering is made ($K=0$), the values are slightly higher than for clustering with $K=1$ and equal to the reference values MS_{orig} given in Table 2.

Table 3 Average Mutation Score in dependence on the clustering parameter K in [%]

Clustering parameter	1. Enterprise Logging		2.Castle		3. Mono-Gendarme	
	O-O	Standard	O-O	Standard	O-O	Standard
1	61.03%	70.18%	63.89%	67.43%	53.57%	69.43%
2	52.88%	63.96%	54.82%	65.20%	43.76%	65.33%
3	49.45%	62.27%	53.08%	63.04%	40.67%	60.82%
4	40.76%	59.79%	46.57%	61.52%	34.39%	54.97%
5	44.30%	59.12%	46.95%	59.27%	34.63%	55.48%
6	38.29%	49.39%	44.30%	59.47%	34.12%	49.40%
7	36.30%	45.48%	42.00%	54.25%	35.00%	46.14%
8	33.92%	44.85%	39.23%	53.88%	31.44%	47.09%
9	30.34%	43.61%	40.72%	55.98%	29.70%	45.77%
10	33.04%	42.98%	37.86%	56.70%	28.60%	42.97%
11	26.22%	37.56%	38.34%	54.21%	24.35%	44.43%
12	30.74%	41.94%	37.50%	53.10%	23.87%	40.88%
13	27.98%	43.77%	37.61%	50.47%	22.30%	39.29%
14	27.80%	34.36%	36.46%	54.42%	26.23%	36.03%
15	28.45%	29.64%	34.79%	48.46%	22.13%	35.72%
16	25.24%	42.82%	36.16%	52.99%	19.42%	35.20%
17	26.01%	39.02%	33.55%	51.76%	20.88%	33.89%
18	28.52%	28.43%	37.06%	51.16%	19.56%	33.71%
19	24.02%	30.04%	33.15%	47.57%	20.16%	33.88%

Quality analysis was aimed at assessing a tradeoff between the decline of mutation score (visible in Table 3) and a possible cost reduction counted in terms of mutant and test number. Based on experiment results, the quality metric EQ (Sec. 3.3) was calculated for different values of the clustering parameter. Table 4 comprises quality values calculated assuming the weight coefficients W_{MS} , W_T , W_M equal to 0.6, 0.2, 0.2 accordingly, i.e. the mutation score accuracy amounts to 60% in the quality measure whereas efficiency factors to 40% (20% for the number of mutants and 20% for the number of tests). The clustering parameter K varies from 0 to 7, as the mutation score was too inaccurate for the higher thresholds.

For OO operators, the best quality of projects 1 and 2 was reached for the clustering parameter $K=1$. This means that mutants in a cluster are killed by test sets including only one different test case. The OO quality of 3rd project was the highest with no clustering, although the quality for $K=1$ was also close to 1.

Table 4 Quality Metrics EQ in dependence of the clustering parameter K

Clustering parameter	1. Enterprise Logging		2.Castle		3. Mono-Gendarme	
	O-O	Standard	O-O	Standard	O-O	Standard
0	0.89	0.73	0.94.	0.89	1.00	0.75
1	1.00	1.00	1.00	0.93	0.97	0.86
2	0.98	0.90	0.73	1.00	0.79	1.00
3	0.82	0.91	0.71	0.87	0.65	0.86
4	0.52	0.98	0.49	0.91	0.47	0.73
5	0.87	0.98	0.55	0.72	0.51	0.80
6	0.51	0.63	0.41	0.84	0.59	0.55
7	0.47	0.47	0.31	0.32	0.67	0.43

Quality metric for standard operators applied to projects 2 and 3 reached maximum when $K=2$. In case of project 1, the maximum is when K equals 1, but other values ($K=2,3,4,5$) gave also good results (above 0.9). It should be noted that for higher values of the parameter ($K=3,\dots,7$) the results of standard operators were in the most cases significantly better (0.1-0.3 higher) than the OO results

While generalizing results, the potential data (mutation score, number of mutants and number of required tests) are compared with the original values without clustering (Table 5). The clustering parameter was assumed to be 1 for OO operators and 2 for standard ones. Results for OO operators averaged for all projects showed that while using 32% of all mutants and 17% of tests, we could obtain 97% of the original mutation score. For standard operators the results of 19% of mutants and 22% of tests could give MS with 91% of the original accuracy.

Table 5 Clustering results for OO and standard mutation

Program		Object-oriented (cluster $K = 1$)			Standard (cluster $K = 2$)		
		Mutation Score [%]	Mutant number	Test number	Mutation Score [%]	Mutant number	Test number
1.Enterprise Logging	Original	61.79%	903	1148	70.92%	1623	1148
	Cluster.	61.03%	295	139	63.96%	221	110
2. Castle	Original	65.82%	1065	642	69.56%	2316	642
	Cluster.	63.89%	333	154	65.20%	681	145
3. Mono-Gendarme	Original	55.91%	855	899	73.86%	4074	899
	Cluster.	53.57%	282	140	65.33%	545	312
Average change [%]		97.2 %	32.3%	17.2%	90.8%	18.8%	22.3%

Time of mutation testing should be decreased when the reduced number of mutants and tests are applied. Effective times of mutant generation and test execution are given in Table 6 and compared with times necessary to generate all mutants

Table 6 Times of mutant generation (including compilation) and of test execution [h:min:sec]

Program	Object-oriented (cluster $K = 1$)		Standard (cluster $K = 2$)	
	Mut. gener. time	Test exec. time	Mut. gener. time	Test exec. time
1 All	06:26:11.2	06:32:36.6	07:22:44.2	11:45:39.2
1 Cluster.	02:07:34.0	00:20:24.0	01:01:44.0	00:07:58.7
2 All	05:37:43.9	07:14:14.2	10:36:59.7	15:44:18.9
2 Cluster.	01:50:54.0	00:56:05.6	01:49:21.0	01:27:42.8
3 All	03:49:32.0	02:02:28.7	13:53:38.9	09:43:36.0
3 Cluster.	01:05:49.0	00:12:45.4	01:54:29.0	00:24:52.2

and execute all mutants with all tests. On average, time of generating a reduced number of mutants took about 30% and 15% of the original time, and time of execution of all test 9% and 5%, for OO and standard mutants respectively.

The programs were quite complex and widely used; however, conclusion validity of experiments is limited due to a small number of programs (three). All programs were of open-source origin that could object external validity.

Mutation score measured with test sets distributed with the projects was low. To alleviate this threat to construct validity additional tests were designed, but the results were still below 100%. Construct validity can also be influenced by metrics and parameter selection. Therefore the analysis was performed for a wide range of K parameter. It also was repeated for another set of weight coefficients: W_{MS} , W_T , W_M equal to 0.8, 0.1, 0.1. In this case mutation score accuracy was more important (0.8) and the quality was maximal when $K=0$ for both OO and standard operators.

6 Conclusions

It was shown, that potential profits of mutation clustering for C# programs could be considerable. While using only 32% or 19% of all mutants and 18% or 22% of tests, the mutation score could be of 97% or 91% close to the original one, for OO and standard mutation operators respectively. In comparison, analogous results for mutant sampling were about 33%, 30% of mutants, 10%, 15% of tests resulting in 85% and 93% of mutation score accuracy [10]. Another method for reduction of mutant and test number - selective mutation gave better accuracy but with a smaller decline of mutant number and analogous number of tests [7].

However, mutation clustering is more difficult to be implemented than selective mutation or mutant sampling, because we generate unnecessary mutants. In a practical approach to clustering, applying statically domain analysis [5], all mutants should be generated but we could benefit from reduced number of test runs with a reduced number of mutants. Concluding, if the potential lowering of

mutation testing complexity and accuracy of mutation results are comparable it would be worthwhile implement methods that are easier to be generalize.

Combining those methods with other approaches to cost reduction, e.g. omitting of redundant mutants [18] or test prioritization [19], remains an open issue.

Acknowledgments. I am very thankful to my student M. Rudnik for extending the CREAM tool and performing mutation testing experiments.

References

- [1] Jia, Y., Harman, M.: An analysis and survey of the development of mutation testing. *IEEE Transactions on Software Engineering* 37(5), 649–678 (2011), doi:10.1109/TSE.2010.62
- [2] Usaola, M.P., Mateo, P.R.: Mutation testing cost reduction techniques: a survey. *IEEE Software* 27(3), 80–86 (2010), doi:10.1109/MS.2010.79
- [3] Hussain, S.: Mutation Clustering. Ms. Thesis, King's College London, Strand, London (2008)
- [4] Jain, A.K., Murty, M.N., Flynn, P.J.: Data Clustering: A Review. *ACM Computing Surveys* 31(3), 264–323 (1999)
- [5] Ji, C., Chen, Z.Y., Xu, B.W., Zhao, Z.H.: A novel method of mutation clustering based on domain analysis. In: *Proc. of 21st Inter. Conf. on Softw. Eng. & Knowledge Eng.*, pp. 422–425 (2009)
- [6] Derezińska, A., Szustek, A.: Object-oriented testing capabilities and performance evaluation of the C# mutation system. In: Szmuc, T., Szpyrka, M., Zendulka, J. (eds.) *CEE-SET 2009. LNCS, vol. 7054*, pp. 229–242. Springer, Heidelberg (2012)
- [7] Derezińska, A., Rudnik, M.: Quality evaluation of object-oriented and standard mutation operators applied to C# programs. In: Furia, C.A., Nanz, S. (eds.) *TOOLS 2012. LNCS, vol. 7304*, pp. 42–57. Springer, Heidelberg (2012)
- [8] Derezińska, A., Szustek, A.: Tool-supported mutation approach for verification of C# programs. In: Zamojski, W., et al. (eds.) *Proc. of Inter. Conf. on Dependability of Computer Systems, DepCoS-RELCOMEX 2008*, pp. 261–268 (2008), doi:10.1109/DepCoS-RELCOMEX.2008.51
- [9] CREAM, <http://galera.i.i.pw.edu.pl/~adr/CREAM/>
- [10] Derezińska, A., Rudnik, M.: Empirical evaluation of cost reduction techniques of mutation testing for C# Programs, Warsaw University of Technology, ICS Res. Rep. 1/2012 (2012)
- [11] Offut, J., Rothermel, G., Zapf, C.: An experimental evaluation of selective mutation. In: *Proc. of 15th Inter. Conf. on Software Engineering*, pp. 100–107 (1993)
- [12] Zhang, L., Hou, S.-S., Hu, J.-J., Xie, T., Mei, H.: Is operator-based mutant selection superior to random mutant selection? In: *Proc. of the 32nd International Conference on Software Engineering, ICSE 2010*, pp. 435–444 (2010), doi:10.1145/1806799.1806863
- [13] Kaminski, G., Praphamontripong, U., Ammann, P., Offutt, J.: A logic mutation approach to selective mutation for programs and queries. *Inform. and Softw. Technol.* 53, 1137–1152 (2011), doi:10.1016/j.infsof.2011.03.009
- [14] Hu, J., Li, N., Offutt, J.: An analysis of OO mutation operators. In: *Proc. of 4th Inter. Conf. Softw. Test. Verif. and Validation Workshops*, pp. 334–341 (2011), doi:10.1109/ICSTW.2011.47

- [15] Mathur, A.P., Wong, W.E.: Reducing the cost of mutation testing: an empirical study. *J. of Systems and Softw.* 31, 185–196 (1995)
- [16] Derezińska, A., Kowalski, K.: Object-oriented mutation applied in Common Intermediate Language programs originated from C#. In: *Proc. of 4th International Conference Software Testing Verification and Validation Workshops*, pp. 342–350 (2011), doi:10.1109/ICSTW.2011.54
- [17] Kryszkiewicz, M.: Fast algorithm finding minima in monotonic Boolean functions, Warsaw Univ. of Technology, ICS Res. Rep. 42/93 (1993)
- [18] Just, R., Kapfhammer, G.M., Schweiggert, F.: Do redundant mutants affects the effectiveness and efficiency of mutation analysis? In: *Proc. IEEE 5th Inter. Conf. on Software Testing, Verification and Validation*, pp. 720–725 (2012), doi:10.1109/ICST.2012.162
- [19] Zhang, L., Marionov, D., Zhang, L., Khurshid, S.: Regression mutation testing. In: *Proc. of Int. Symp. on Software Testing, ISSTA 2012*, pp. 331–341 (2012)

Using Virtualization Technology for Fault-Tolerant Replication in LAN

Fernando Dettoni¹, Lau Cheuk Lung¹, and Aldelir Fernando Luiz²

¹ Departamento de Informática e Estatística,
Universidade Federal de Santa Catarina,
Florianópolis, Brazil
{fdettoni, lau.lung}@inf.ufsc.br

² Departamento de Automação e Sistemas,
Universidade Federal de Santa Catarina, Florianópolis, Brazil
aldelir@das.ufsc.br

Abstract. We present an architecture and an algorithm for Byzantine fault-tolerant state machine replication. Our algorithm explores the advantages of virtualization to reliably detect and tolerate faulty replicas, allowing the transformation of Byzantine faults into omission faults. Our approach reduces the total number of physical replicas from $3f+1$ to $2f+1$. Our approach is based on the concept of twin virtual machines, where there are two virtual machines in each physical host, each one acting as a failure detector of its twin.

1 Introduction

More and more, computing systems are being used in critical systems and have to operate correctly even under the presence of faults. These faults can be accidental, like crash faults, or arbitrary, called Byzantine [1]. Thus, to ensure that these systems remain available under fault conditions, it is necessary to develop Byzantine fault-tolerant (BFT) mechanisms. One of the most used architecture is the State Machine Replication (SMR)[2], using deterministic state machines to offers a replicated service. Many BFT SMR-based approaches were developed (e.g. [3], [4], [5]). Among these, the PBFT [3] is often considered to be a baseline, being the first practical BFT algorithm and having most of later approaches derived from it.

Another technique consists of using unreliable failure detectors [6]. Despite of supporting at first just crash faults, some proposals were able to extend the idea to support Byzantine faults [7][8]. These fault detectors help the system giving some hints about replicas appearing to be faulty.

The virtualization can also be considered a Byzantine fault-tolerant technique, because it introduces an isolation level between the virtual machines. Several approaches use virtualization to protect some components from others' failure (or intrusion) [9], [10]. Virtualization techniques are widely accepted by industry, and

are largely used, e.g., by cloud computing services as Amazon Web Services and Windows Azure.

The PBFT and many other BFT SMR-based algorithms have a high implementation cost having normally the resiliency of $n \geq 3f + 1$, i.e., need $n > 3f$ replicas to tolerate f faulty replicas. To diminish this cost, some approaches emerged using a trusted component to limit the behavior of faulty replicas using only $n \geq 2f + 1$ replicas [11], [12], [13]. Other approaches were proposed running only $f + 1$ replicas, but keeping more $2f$ replicas waiting in a paused state without consuming any CPU time until it is activation [14].

We present a new efficient BFT SMR-based architecture based on virtualization, called TwinBFT. We reduced the number of required physical machines n from $n \geq 3f + 1$ to $n \geq 2f + 1$ to tolerate f faults. Furthermore, we reduced the number of communication steps in the normal case from 5 (in PBFT) to 3, without the client participation in the agreement. To our knowledge, this is the first algorithm with this number of steps without using a speculative approach [5], [13], involving the participation of the client in the agreement.

The proposed approach consists of use a set of twin virtual machines, executing the same application service in each one of the $N \geq 2f + 1$ physical replicas. We use a set with only two virtual machines. Every virtual machine executes the same service, and each set acts as a replica of the state machine replication. The main idea is to use each virtual machine as a failure detector to its twin: upon sending a request to a pair of twin virtual machines, both must provide the same answer, otherwise the whole physical machine hosting the twins is considered faulty and the messages sent by this replica are ignored by the others. This way, each set of twin virtual machines will either act correctly or omit messages. This omission, however, are tolerated by the state machine replication. A crash fault too, is a kind of omission and, therefore, is also tolerated.

The proposed architecture do not intends to offer the ideal solution too all the cases. The use of virtual machines is suitable to companies opened to this kind of technology, as cloud computing companies. It is also recommended when you do not have trusted components or cannot wait for the activation of sleeping replicas in the case of failure. In these cases, an efficient BFT SMR with only $2f + 1$ physical replicas ($4f + 2$ virtual machines) using virtualization may be suitable.

In Section 2, we give an overview of some related work showing the state of the art. Then, Section 3 describes our system model and assumptions. A detailed explanation of the algorithm is given in Section 4. Following, the Section 5 presents an evaluation of algorithm and the Section 6 summarize our conclusions.

2 Related Work

Several works were proposed recently in BFT to produce Byzantine fault-tolerant services based on SMR. Being the first practical approach for BFT protocols, PBFT is one of the most successful SMR protocol [3]. Although being practical, the cost for implementing PBFT is quite high, requiring at least 4 replicas ($3f + 1$ to

$f = 1$) and 5 communication steps. Thus, numerous approaches derive from it with two goals: reduce the number of required replicas and improve the performance.

Many works offered an alternative to improve the resiliency of PBFT reducing the number of replicas. Yin et al. introduces an architecture separating the service in two distinct layers, one responsible for agreement with $3f+1$ replicas and another one executing the requests with only $2f + 1$ replicas [4]. While still need $3f + 1$ replicas, the execution replicas are likely to be much more expensive than agreement replicas.

Correia et al. presented the first solution to execute a BFT SMR with only $2f + 1$ replicas, using a trusted distributed component [11]. After this, another work showed the first algorithm of it is kind based in a trusted local component using the abstraction of attested append-only memory [12]. Veronese et al. [13] proposed two algorithms using a trusted component, which supply unique identifiers for each message. The first one, MinBFT, reduced the number of necessary replicas to $2f + 1$ and the number of communication steps to 4. The second, a speculative version called MinZyzyva, reduced the communication steps even further, to 3, keeping the number of replicas in $2f + 1$.

In another approach, Stumm et al. [15] takes advantage of virtualization techniques to reduce the number of required replicas to $2f + 1$ since the VMM provides a secure communication between the replicas. This approach, however, require all replicas running on the same physical host and, therefore, do not tolerates crash faults on the physical machine, unlike this paper.

Another work, also based on the idea of two replicas watching each other, is presented in [16] using a signal-on-fail approach. This approach needs $4f+2$ physical machines and requires a synchronous and trusted communication between each pair of replicas, which is a difficult assumption to be guaranteed in practice. In this same line of thought Inayat and Ezhilchevan presented an optimist multicast BFT protocol with total order based on the signal-on-fail approach. The authors showed that in a normal execution, it is likely to have better performance than other BFT approaches [17].

Several other works presented solutions to improve the performance of PBFT. Kotla et al. presented Zyzyva, an algorithm able to reduce the number of communication steps in the absence of faults [5]. Instead of trying to reach an agreement before sending the reply to the client, the service replies speculatively. The service needs to execute the request again and reach an agreement only if the replies received by the client differ from each other. This approach takes advantage of most cases of the execution is free of failure but requires the ability to revert operations to ensure consistency in case of failures.

As stated in the Introduction, this paper explores another point of the design space using virtualization to deploy a BFT state machine replication with only $2f + 1$ physical replicas (and $4f+2$ virtual machines) and only 3 communication steps in the normal case of execution.

Several works use virtualization to isolate components of software. Two of the first use virtualization to protect the intrusion detector from the intruders [10], and a more recent one uses the same idea to protect a honeypot monitor [9].

Nevertheless, the hypervisor security is mandatory to obtain isolation and some works studied how to improve this security. Murray et al. proposed dissociates the virtualization system as a solution to lower the trusted computing base of the system [18]. The NoHype goes further removing the hypervisor of the way and executing the virtual machines natively [19]. The HyperSafe uses another approach: protects the hypervisor detecting attacks that modify the control flux, as buffer overflows [20].

3 System Model

The architecture of the system is presented in Figure 1. The system is composed by a set of n physical machines (e.g. host) $H = \{h_1, h_2, \dots, h_n\}$ where $n \geq 2f + 1$ and f is the maximum number of faulty physical machines at any time. Each host of the Figure 1 contains a VMM (*Virtual Machine Manager*) with two virtual machines, called twins virtual machines, running one process each. Both servers $\{s_i, s_i'\}$ execute the same service (with different versions because of software diversity), and communicate between each other to validate each message before send to other processes.

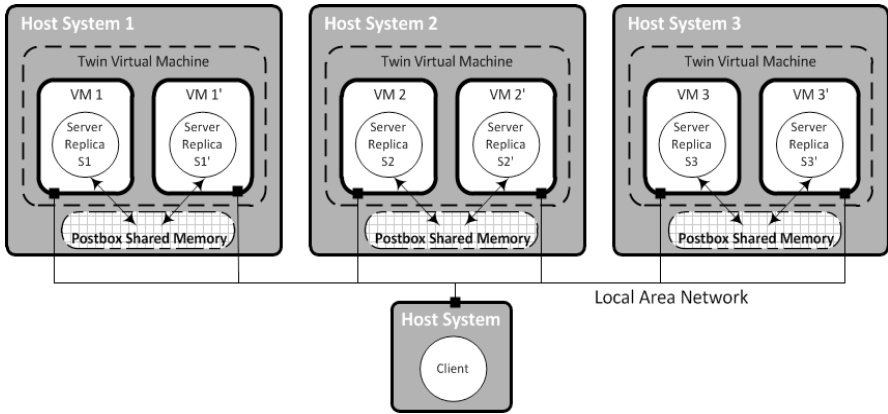


Fig. 1 TwinBFT - Twin virtual machines architecture

We assume at most f virtual machines acting Byzantine, but no more than 1 is faulty in the same physical host. When a virtual machine is arbitrarily faulty, the validation mechanism transforms this fault to an omission fault. Therefore, we assume up to f physical hosts can be faulty by crash or omission, accidentally or due to a failure in one of its virtual machines. To substantiate the f faults limit we have to rely in software diversity techniques, i.e., different implementations of the process at each replica in a physical host [21], [22]. This diversity reduces the chance of more than one virtual machine at the same physical host being attacked

simultaneously. We assume the virtualization provides isolation between the virtual machines and the VMM / Hypervisor.

No assumptions are made about the time needed to compute a request. The communication between different *VMs* inside the same host is made through a shared memory space, called postbox. The processes at different hosts communicate through the local network, by message passing only. This local network can fail to deliver messages, delay, or duplicate messages.

Each host can assume two different roles: (1) primary host, which is responsible for defining the order for executing the client requests; and (2) backup host, which executes the requests following the order proposed by the primary. Within a primary host, a process can assume two possible roles: (1) leader, which is responsible for assign the sequence number for client's requests; and (2) follower, which executes the requests following the order defined. All the processes within backup hosts are considered followers. The primary host h_i are defined by $i = v \bmod |S|$ where v is the current view. The primary leader process within a server is, by definition, s_i .

We use cryptographic techniques to authenticate messages and ensure authenticity of messages [24]. Each pair of processes share among each other a secret key used to generate a MAC (*Message Authentication Code*) vector [3] with a valid signature for each process.

We assume each physical machine with only two virtual machines (*VMs*) where, for a given input, the replies supplied by both should be the same to the physical machine to not be considered faulty.

4 Algorithm

The service in our algorithm is modeled as a state machine replication, distributed across the service nodes. The replicas move through a succession of configurations called views. In each view, we have one primary replica, which is responsible for defining the message order, and forward the request to all service replicas. As stated by [2], the state machine must be deterministic, and all replicas must start in the same state, otherwise the safety cannot be guaranteed.

In this section we will discuss our proposal of transformation through a well-known approach. In this sense, we present an adaptation of PBFT protocol [23] to our proposed model. In each view, only one replica s_j is the primary (or leader), which is responsible for defining the message order and forward the request to all service replicas. If a message sent by any replica is signed by both *VMs* in a host we assume this message as correct, since we assume that only one *VM* can fail at the same time in the same host.

4.1 Properties

As a state machine replication, our algorithm must ensure the following properties to provide a correct service:

- **Total Order** (*safety*): a request is executed sequentially and at the same order on every replica, i.e. despite replication, the operations are atomically executed in order and the system behaves as a centralized one;
- **Termination** (*liveness*): a request issued by a client eventually complete, regardless of failure.

Our algorithm provides both safety and liveness, assuming that no more than $f = (n - 1)/2$ hosts are faulty and there is at least one correct process s in each host. To ensure that all replicas will execute the requests in the same order, all the replicas follow the order defined by the leader and the leader can be assumed correct if both processes at the primary host sign the order proposed by the leader. Our protocol ensures safety regardless of timing but to ensure liveness we need to make some assumptions about synchrony and message loss.

To ensure that all replicas execute the same requests in the same order, all replicas follow the order defined by primary leader. A consensus algorithm is not necessary because we can trust the order defined by the primary leader, provided that our previous assumptions are not violated. This is because after the primary leader defines the order, this order will not be accepted by the other replicas without the agreement of both processes in primary host.

4.2 Description of Algorithms

In this section, we will discuss our algorithms in detail. First, we show in Figure 2 a diagram-based view of our main algorithm, to make easy to understand it. This architecture assumes $f = 1$, where three hosts are required, each one with two VMs. Each pair of VMs inside the same host communicates between each other through a trusted FIFO channel called postbox. The postbox can be faster than the network by using a shared memory abstraction provided by the VMM.

The algorithm works basically as follows:

1. Client issues a request to both VMs in primary host;
2. The primary leader s_i define a sequence number and post an “ORDER” message on postbox;
3. The primary follower s_i' reads the message from postbox, gets the sequence number and posts on postbox an “ORDER” message with the sequence number received;
4. Both VMs sign the “ORDER” read from twin and send to backup replicas;
5. As soon as each VM inside a backup replica receive the message “ORDER”, they execute the operation, and post a signed “REPLY” on the postbox;

6. When a VM reads a “REPLY”, it compares with the one computed locally and if all fields matches, attaches its own signature to the message and send to client;
7. If the client receives at least $f + 1$ correctly signed (by both twin VMs) replies from distinct physical replicas, it accepts the result.

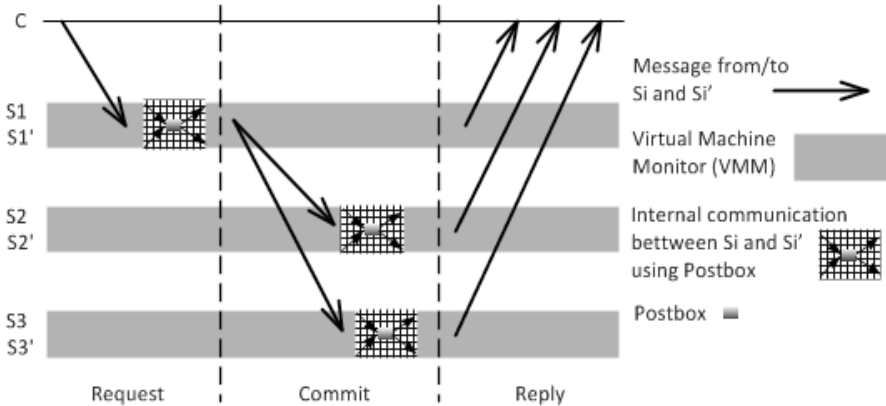


Fig. 2 Algorithm's steps sequence in normal case operation with $f = 1$

In a normal execution, a client sends a request to the service and waits until receives at least $f + 1$ valid replies from distinct replicas. The request message has the form $\langle \text{REQUEST}, c, seq, op \rangle_{oc}$ where c is the client id, seq is a request id on a client, and op is the operation to be executed on the service. If the client does not receive $f + 1$ messages soon enough, it multicast the request to all replicas.

4.3 Normal Case Operation

The algorithm, at normal case operation, running in each one of the replicas has two concurrent tasks. The Task 1 is responsible for reading the messages received through network. Task 2 is responsible for reading the messages from postbox, posted by its twin. The state of each process is composed by the state of the service, a message buffer and the current view. This state is shared among the tasks.

When any of processes $\{s_i, s_i'\}$ in primary host receives a request from client, s_i generates a new sequence number n and create a message $\langle \langle \text{ORDER}, s_i, v, n, dm \rangle_{opi}, m \rangle$ where v is the current view number, and dm is the digest of message m . As soon as s_i' reads the s_i “ORDER” message from the postbox and have the “REQUEST” message in the message buffer, it gets the sequence number proposed by s_i , creates an “ORDER” message and posts on the postbox. When each one reads an “ORDER” message from postbox, it verifies if all parameters correspond to the ones computed locally and, if yes, add it is own signature to the twin message and multicast to backup replicas.

To any “ORDER” message received, the replicas will consider it valid if:

- Message is correctly signed, i.e., if received from network both twin *VM* machines on the replica must sign it, and if received from postbox, its own twin machine must sign it;
- The view in the message is the current view;
- Has not accepted another “ORDER” message with the same sequence number for a different request.
- The sequence number is between a low and high water marks h and H (in practice, if this verification is made when the primary follower reads the “ORDER” message from postbox, a backup replica will never receive a message outside these water marks).

Upon the receiving of “ORDER” message by both twin processes on a physical host, each one of $\{s_i, s_i'\}$ verifies if the message is valid and, if yes, it executes the operation and create a message $\langle \text{REPLY}, si, v, seq, c, res \rangle_{opi}$ where res is the result of executing the operation, and posts on the postbox. Once the “REPLY” has been read from the postbox, it compares each parameter of the message and, if all parameters are identical to the ones locally computed then sign the message generated by its twin and send to client.

When the client receives a “REPLY” it accepts as a valid message if the following conditions are true:

- Is signed by both processes $\{s_i, s_i'\}$ on sender host;
- It has not accepted yet a valid message from any of the twin processes on the sender host.

The client waits until have received at least $f + 1$ valid messages from the replicas to accept the result. If it could not receive these messages, soon enough, it multicast the “REQUEST” to all replicas.

5 Analytic Evaluation

In Table 1 we can see a comparison between our approach and state-of-the-art BFT algorithms in the literature. All numbers considers only gracious executions. The benefits of using a twin machines approach are visible on the number of replicas and communication steps. While our approach has the lowest number of replicas, along with [11], [12], [13], it has the same number of communication steps of the speculative algorithms [5], [13] even in case of faults, this is an interesting achievement. Speculative algorithms, however, require more communication steps in case of faults, additionally involving the client in the protocol, leaving behind the transparency of the protocol.

Table 1 Comparison of Evaluated BFT Algorithms

	Number of Replicas	Number of Processes	Number of Physical Machines	Communication Steps (latency)	Speculative / Optimist
PBFT [3]	3f+1	3f+1	3f+1	5	no
Zyzyva [5]	3f+1	3f+1	3f+1	3 / 5	yes
TTCB [11]	2f+1	2f+1	2f+1	5	no
A2M-PBFT-EA [12]	2f+1	2f+1	2f+1	5	no
MinBFT [13]	2f+1	2f+1	2f+1	4	yes
TwinBFT	2f+1	4f+2	2f+1	3	no

As our approach uses two virtual machines at each replica, we have a bigger number of processes despite of the number of required physical machines being the same as [11], [12], [13].

6 Conclusions

By exploring some virtualization techniques, we proposed a less expensive alternative algorithm to BFT. We show that is possible to implement a reliable $2f + 1$ SMR algorithm in an asynchronous environment. Despite of relying in a secure communication between each pair of virtual machines, we believe that virtualization is widely available today and can provide a good isolation between the replicas and the external world. Moreover, we reduced the number of necessary communication steps, reducing the cost of communication.

References

- [1] Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.* 4(3), 382–401 (1982)
- [2] Schneider, F.B.: Implementing fault-tolerant services using the state machine approach: a tutorial. *ACM Comput. Surv.* 22(4), 299–319 (1990)
- [3] Castro, M., Liskov, B.: Practical Byzantine fault tolerance. In: *Proc. of the 3rd OSDI*, pp. 173–186. USENIX Association, Berkeley (1999)
- [4] Yin, J., Martin, J.P., Venkataramani, A., et al.: Separating agreement from execution for Byzantine fault tolerant services. *SIGOPS Oper. Syst. Rev.* 37, 253–267 (2003)
- [5] Kotla, R., Clement, A., Wong, E., et al.: Zyzyva: speculative Byzantine fault tolerance. *Commun. ACM* 51, 86–95 (2008)
- [6] Chandra, T.D., Toueg, S.: Unreliable failure detectors for reliable distributed systems. *J. ACM* 43(2), 225–267 (1996)
- [7] Doudou, A., Garbinato, B., Guerraoui, R., Schiper, A.: Muteness failure detectors: Specification and implementation. In: Hlavicka, J., Maehle, E., Pataricza, A. (eds.) *EDDC 1999*. LNCS, vol. 1667, pp. 71–87. Springer, Heidelberg (1999)
- [8] Kihlstrom, K.P., Moser, L.E., Melliar-Smith, P.M.: Byzantine fault detectors for solving consensus. *The Computer Journal* 46 (2003)

- [9] Jiang, X., Wang, X.: “Out-of-the-box” monitoring of VM-based high-interaction honeypots. In: Kruegel, C., Lippmann, R., Clark, A. (eds.) RAID 2007. LNCS, vol. 4637, pp. 198–218. Springer, Heidelberg (2007)
- [10] Garfinkel, T., Rosenblum, M.: A virtual machine introspection based architecture for intrusion detection. In: Proc. of the Network and Distributed Systems Security Symposium (2003)
- [11] Correia, M., Neves, N.F., Verissimo, P.: How to tolerate half less one Byzantine nodes in practical distributed systems. In: Proc. of the 23rd IEEE SRDS, pp. 174–183 (2004)
- [12] Chun, B.G., Maniatis, P., Shenker, S., et al.: Attested append-only memory: making adversaries stick to their word. In: Proc. of the 21st ACM SOSP, pp. 189–204 (2007)
- [13] Veronese, G.S., Correia, M., Bessani, A.N., et al.: Efficient Byzantine fault tolerance. *IEEE Transactions on Computers* 62(1), 16–30 (2013)
- [14] Wood, T., Singh, R., Venkataramani, A., et al.: ZZ and the art of practical BFT execution. In: Proceedings of the 6th ACM SIGOPS/EuroSys European Systems Conference, pp. 123–138 (2011)
- [15] Stumm, V., Lung, L.C., Correia, M., et al.: Intrusion tolerant services through virtualization: A shared memory approach. In: Proc. of the 24th IEEE AINA, pp. 768–774 (2010)
- [16] Mpoeleng, D., Ezhilchelvan, P., Speirs, N.: From crash tolerance to authenticated Byzantine tolerance: A structured approach, the cost and benefits. In: Proceedings of the IEEE/IFIP 33rd International Conference on Dependable Systems and Networks, pp. 227–236 (2003)
- [17] Inayat, Q., Ezhilchelvan, P.: A performance study on the signal-on-fail approach to imposing total order in the streets of byzantium. In: Proc. IEEE DSN, pp. 578–587 (2006)
- [18] Murray, D.G., Milos, G., Hand, S.: Improving Xen security through disaggregation. In: Proceedings of the 4th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 151–160 (2008)
- [19] Szefer, J., Keller, E., Lee, R.B., et al.: Eliminating the hypervisor attack surface for a more secure cloud. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, pp. 401–412 (2011)
- [20] Wang, Z., Jiang, X.: HyperSafe: A lightweight approach to provide lifetime hypervisor control-flow integrity. In: Proc. of the IEEE Security and Privacy Symposium, pp. 380–395 (2010)
- [21] Bessani, A., Daidone, A., Gashi, I., et al.: Enhancing fault / intrusion tolerance through design and configuration diversity. In: Proceedings of the 3rd Workshop on Recent Advances on Intrusion-Tolerant Systems (2009)
- [22] Gashi, I., Popov, P.T., Strigini, L.: Fault tolerance via diversity for off-the-shelf products: A study with SQL database servers. *IEEE Transactions on Dependable and Secure Computing* 4(4), 280–294 (2007)
- [23] Castro, M., Liskov, B.: Authenticated Byzantine fault tolerance without public-key cryptography. Technical report, Cambridge, MA, USA (1999)
- [24] Wangham, M.S., Lung, L.C., Westphall, C.M., da Silva Fraga, J.: Integrating SSL to the JACOWEB security framework: Project and Implementation. In: IM 2001, pp. 779–792 (2001)

Quantification of Simultaneous-AND Gates in Temporal Fault Trees

Ernest Edifor, Martin Walker, and Neil Gordon

Department of Computer Science, University of Hull,
Cottingham Road, Hull, HU6 7RX, UK
{e.e.edifor@2007,martin.walker@,n.a.gordon@}hull.ac.uk

Abstract. Fault Tree Analysis has been a cornerstone of safety-critical systems for many years. It has seen various extensions to enable it to analyse dynamic behaviours exhibited by modern systems with redundant components. However, none of these extended FTA approaches provide much support for modelling situations where events have to be "nearly simultaneous", i.e., where events must occur within a certain interval to cause a failure. Although one such extension, Pandora, is unique in providing a "Simultaneous-AND" gate, it does not allow such intervals to be represented. In this work, we extend the Simultaneous-AND gate to include a parameterized interval – referred to as *pSAND* – such that the output event occurs if the input events occur within a defined period of time. This work then derives an expression for the exact quantification of *pSAND* for exponentially distributed events and provides an approximation using Monte Carlo simulation which can be used for other distributions.

1 Introduction

The effects of technology now pervade almost every sphere of life, increasing human dependency on them. The failures of some of these systems can have devastating effects on human life and the environment. Such systems with catastrophic effects are known as high consequence or safety-critical systems, so assessing their reliability is of increasing importance. Some modern systems depend on duplicated or stand-by components to improve their reliability. However, this feature poses other challenges for system designers who need to model and evaluate system reliability appropriately.

Fault Tree Analysis (FTA) is a popular technique for analysing how faults or combinations of them can cause the total failure, also known as the top-event, of a system or subsystem. Developed in the 1960s, fault trees can be analysed logically (qualitatively) or probabilistically (quantitatively). Traditionally, these analyses are mostly done with the Boolean AND and OR gates. The end products of the logical analysis are the Minimal Cut Sets (MCSs), which are combinations of basic faults necessary and sufficient to cause the occurrence of the top-event. The probabilistic analysis produces numbers representing the probability of the

top-event occurrence or the importance of the various terms in the MCSs in relation to their contribution to the top-event probability.

However, classical FTA is considered static [1-2], which is a significant disadvantage in modern dynamic systems: there are often occasions where accurately representing the failure behaviour of the system requires a more flexible and in-depth description than that provided by ordinary AND and OR gates. Not doing so can result in inaccurate evaluation of MCSs and estimation of the top-event probability [3-5]. As a result, FTA has seen various modifications to enable it to model and evaluate modern systems presenting dynamic behaviours. A popular and widely used solution is the Dynamic Fault Tree (DFT) [4]. DFTs make use of a pre-existing definition of Priority-AND (PAND) gates [6] and introduce other dynamic gates -- Spare, Functional Dependency (FDEP) and Sequential Enforcing (SEQ) -- to model and evaluate fault trees with dynamic features.

Apart from DFTs, FTA has seen other modifications. A recent modification of FTA is Pandora [5] [7-8] which analyses fault trees logically with three temporal gates – PAND, POR, and SAND. PAND stands for Priority-AND and it occurs if and only if an input event occurs strictly before another input event; inputs are arranged left-to-right with the leftmost occurring first. POR is for Priority-OR – which represents the situation where an output event occurs if its first input event occurs before its second input event or just the first input event occurs without the occurrence of the second input event. Finally, SAND stands for Simultaneous-AND. A SAND gate is used to represent the situation where all input events to an output event occur at the same time.

Pandora analyses fault trees with its temporal gates by use of its novel temporal laws [5] to generate Minimal Cut Sequences (MCSQs), analogous to Minimal Cut Sets; this enables a form of temporal qualitative analysis. MCSQs represent combinations or sequences that are sufficient and necessary to cause the top event. There are also techniques for the quantitative evaluation of the AND and OR gates [9], PAND gate [6] [10-12] and POR [13]. Taking an ideal situation, the SAND gate evaluates to zero [14] because the probability of exponentially distributed, independent events occurring simultaneously is zero.

One situation which is not covered by either DFTs or Pandora is that of "near simultaneity". In some scenarios we may wish to differentiate between a failure that occurs because two (or more) events occur within a given time of each other, and a different failure that occurs when those events occur further apart. This kind of 'interval' occurrence is comparatively common. For example, if a fire is detected and the sprinklers activate almost immediately to extinguish it, then the damage may be comparatively minimal; conversely, if there is a delay between the alarm and the sprinklers, perhaps because of a blockage in a pipe, the damage may be significantly worse.

Approaches to formalise such scenarios include Duration Calculus [15-16], PLTLP [17], CSDM [18], CTL [19], simplified CTL* [20] and PFTTD [21]. In general, these approaches are intended to formalise the semantics of timing and

sequences of events in the context of simulation and formal specification. Only CSDM and Durational Calculus are intended to work with fault trees, and even this is primarily via an initial transformation to other forms, e.g. Petri Nets.

By contrast, DFTs and the Pandora-based approach in this paper are both fault tree-centric approaches designed with analysis, rather than specification, as the primary goal. However, DFTs in general lack good support for qualitative analysis, so we focus here on extending Pandora's existing capabilities to solve this issue.

Pandora's SAND gate provides the closest semantics to this scenario, but needs extending so that it can model a slight time delay between the input events. Therefore this work seeks define a delay-inclusive SAND gate, which is hereafter referred to as the parameterized SAND (pSAND), and to define means of probabilistically evaluating such a gate. The new gate is defined in section 2, and in section 3 we present two new mathematical techniques for evaluating the pSAND gate: Calculus (exact solution) and Monte Carlo Simulation (approximate solution). Both techniques are applied to a small case study in section 4. Discussions of the results and evaluation of proposed techniques are made in sections 5 and in section 6 we present our conclusions.

2 Parameterized SAND (pSAND) Description

Nearly simultaneous events are events that will trigger the occurrence of an output event if they should happen within a relatively short period of time – i.e., within a given *interval*. A classic example is discussed in the case study in Section 4. To model and evaluate the pSAND gate where input events occur within an interval, it is expedient that its symbols for modelling, qualitative and quantitative analysis be altered to accommodate the change. Therefore the original SAND gate is not redefined entirely, but rather slightly extended:

Semantics of pSAND: All input events of the pSAND gate must occur and they must do so within a relatively short interval of duration ' d ', which starts with the first input event to occur. The pSAND is therefore false if any of its inputs do not occur or if they occur outside the interval, i.e., the time between the first and last input to occur is more than d .

Since it is not the focus of this work to completely redefine the SAND gate, its original graphical symbol (Fig. 1a) is retained for the situation where $d=0$: input events occur at exactly the same time. Alternatively, Fig. 1b can also be used to represent the same scenario where $d=0$. Fig. 1c on the contrary represents a situation where the input events are nearly simultaneous with a duration, d , between them and $d>0$.

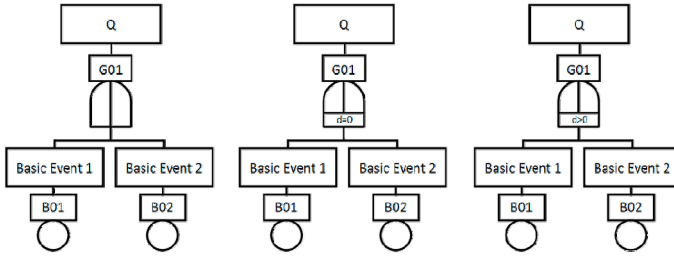


Fig. 1 pSAND graphical representations

SAND’s abbreviation and symbol are changed to pSAND and $\&_d$ respectively, where d will be the duration of the interval. In addition, expressing $A \&_d B$ will be represented as $A \&_d B \{t_0, t_1\}$ where $d > 0$ and $d = t_1 - t_0$; t_0 being the time of beginning the duration and t_1 the time of end of the interval within which the output event becomes true.

For any two independent events A and B, Fig. 2 represents the timing behaviour for $A \&_d B$ and equations a-e their corresponding mathematical expressions.

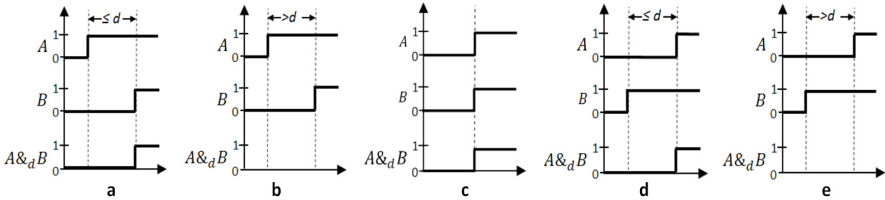


Fig. 2 pSAND timing behaviour

$$\begin{aligned} \{t(A) < t(B)\} \text{ AND } \{t(B) - t(A) \leq d\} &\rightarrow t(A \&_d B) = t(B) & (a) \\ \{t(A) < t(B)\} \text{ AND } \{t(B) - t(A) > d\} &\rightarrow t(A \&_d B) = \emptyset & (b) \\ \{t(A) = t(B)\} &\rightarrow t(A \&_d B) = t(A) & (c) \\ \{t(A) > t(B)\} \text{ AND } \{t(A) - t(B) \leq d\} &\rightarrow t(A \&_d B) = t(A) & (d) \\ \{t(A) > t(B)\} \text{ AND } \{t(A) - t(B) > d\} &\rightarrow t(A \&_d B) = \emptyset & (e) \end{aligned}$$

In Fig. 2a A occurs before B and the duration of delay between them is less than or equal to d but greater than zero: thus $A \&_d B$ occurs/becomes true. In Fig. 2b A occurs before B within the duration which is greater than d and d is greater than zero: $A \&_d B$ does not occur. In Fig. 2c, if A and B occurred at exactly the same time then the pSAND would still be true; however, the probability of this occurring with independent events is essentially 0 (though if A and B were not independent, the probability could be non-zero). In Fig. 2d B occurs before A within the duration which is less or equal to d but greater than zero: $A \&_d B$ occurs. In Fig. 2e

B occurs before A with a duration of delay which is greater than d and greater than zero: $A \&_d B$ does not occur. pSAND occurs when all of its input events occur within a specified interval of duration.

3 Mathematical Model

It is assumed that all events are non-repairable, exponentially distributed with constant failure rates and any system under study is coherent with $F(X)$ being the cumulative distributive function of X . Fig. 3 is a graphical representation of a pSAND scenario with two input events E_1 and E_2 having constant failure rates λ_1 and λ_2 respectively and a delay ‘ d ’ between the occurrence of E_1 and E_2 ; E_1 occurring at ‘ t_0 ’ and E_2 occurring anytime between ‘ t_0 ’ and ‘ t_1 ’.

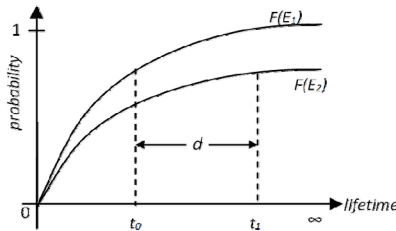


Fig. 3 pSAND Mathematical graph for two events

The probability of E_1 occurring at t_0 and E_2 occurring anytime between t_0 and t_1 is the pSAND probability of E_1 and E_2 . pSAND is commutative therefore event sequence does not matter. Algebraically this can be expressed as:

$$E_1 \&_d E_2 \{t_0, t_1\} = E_2 \&_d E_1 \{t_0, t_1\} = E_1 \{0, t_0\} * E_2 \{t_0, t_1\} + E_2 \{0, t_0\} * E_1 \{t_0, t_1\} \quad (1)$$

$$Pr(E_1 \&_d E_2) \{t_0, t_1\} = Pr(E_1 \{0, t_0\} * E_2 \{t_0, t_1\} + E_2 \{0, t_0\} * E_1 \{t_0, t_1\}) \quad (2)$$

$$= Pr(E_1 \{0, t_0\} * E_2 \{t_0, t_1\}) + Pr(E_2 \{0, t_0\} * E_1 \{t_0, t_1\}) - Pr((E_1 \{0, t_0\} * E_2 \{t_0, t_1\}) * (E_2 \{0, t_0\} * E_1 \{t_0, t_1\})) \quad (3)$$

However, $Pr((E_1 \{0, t_0\} * E_2 \{t_0, t_1\}) * (E_2 \{0, t_0\} * E_1 \{t_0, t_1\}))$ equals 0, assuming E_1 and E_2 are independent and a continuous model of time is used, therefore,

$$Pr(E_1 \&_d E_2) \{t_0, t_1\} = Pr(E_1 \{0, t_0\} * E_2 \{t_0, t_1\}) + Pr(E_2 \{0, t_0\} * E_1 \{t_0, t_1\}) \quad (4)$$

$$= Pr(E_1 \{0, t_0\}) * Pr(E_2 \{t_0, t_1\}) + Pr(E_2 \{0, t_0\}) * Pr(E_1 \{t_0, t_1\}) \quad (5)$$

$$= Pr(E_1 \{0, t_0\}) * Pr(E_2 \{t_0, t_1\}) + Pr(E_2 \{0, t_0\}) * Pr(E_1 \{t_0, t_1\}) \quad (6)$$

$Pr(X\{a,b\}) = F(X)\{a,b\} = \text{Exp}(-a*x) - \text{Exp}(-b*x)$, $F(X)$ being the Cumulative Distributive Function (CDF) of X . Therefore,

$$Pr(E_1 \&_d E_2) \{t_0, t_1\} = F(E_1) \{0, t_0\} * (F(E_2) \{t_0, t_1\}) + F(E_2) \{0, t_0\} * (F(E_1) \{t_0, t_1\}) \quad (7)$$

Given that, $Pr(X\{t_0, t_1\}) = Pr(X\{0, t_1\}) - Pr(X\{0, t_0\})$

$$Pr(E_1 \&_d E_2 \dots \&_d E_{n-1} \&_d E_n)\{t_0, t_1\} = F(E_1)\{0, t_0\} * (F(E_2)\{0, t_1\} - F(E_2)\{0, t_0\}) + F(E_2)\{0, t_0\} * (F(E_1)\{0, t_1\} - F(E_1)\{0, t_0\}) \tag{8}$$

It follows that for any n independent events $E_1, E_2, \dots, E_{n-1}, E_n$ the pSAND probability is

$$Pr(E_1 \&_d E_2 \&_d \dots \&_d E_{n-1} \&_d E_n)\{0, t_0, t_1\} = \sum_{i=1}^n \left(F(E_i)\{0, t_0\} * \left(\prod_{\substack{j=1 \\ j \neq i}}^n F(E_j)\{t_0, t_1\} \right) \right) \tag{9}$$

Monte Carlo (MC) simulation is used to understand and control complex stochastic real world systems. It has been employed in weather forecasting, insurance, engineering, financial market, chemical processes, telecommunication networks and the like. Its popularity in reliability engineering has increased over the past decade. It has been employed in qualitative [22–23] and quantitative [10] analysis.

A typical MC implementation commences by mathematically modelling known and unknown variables of the system under study. This model is run a large number of times – called trials – based on the outputs expected from the model with randomised values of appropriate input variables. This simulated random behaviour of the system model allows the estimation of complex real world probabilities provided it has been well modelled and randomised an appreciable number of times.

An algorithm to estimate the pSAND probability for n independent events using Monte Carlo simulation is:

1. Generate random numbers for the failure rates of all events.
2. If, for all events, the first event occurs between $\{0, t_0\}$ while all other events occur between $\{t_0, t_1\}$ then keep count
3. Repeat steps 1 and 2 for a large number of trials.
4. Evaluate the pSAND probability by dividing the number of counts by the trials.

4 Case Study

The pSAND is useful in differentiating the effects of failures that occur within a small interval from effects of more widely spaced failures. To demonstrate the relevance of pSAND and prove the proposed techniques, in Fig. 4 we present an automotive brake-by-wire (BBW) system, which is adapted from [8].

The BBW system contains individual brake actuators with rotation sensors connected at each of its four wheels. The central bus serves two major functions: it is a medium for controlling the actuators and also carries the signals from the sensors. The signals from the sensors are inputs to a pair of Electronic Control Units (ECUs) that control the brakes. To prevent inadvertent braking caused by an error in one ECU, the comparator determines the output of both ECUs; if they agree, commands are sent to the actuators to activate braking. 'Vehicle dynamics'

is a virtual component representing the effect the brakes have on the handling of the vehicle and can be thought of as the output of the braking system. It should be emphasized that the vehicle dynamics is not a physical component of the system but rather a way of representing the success (or failure) of the braking effect on the vehicle. For the sake of demonstrating the pSAND concept, the case study presented in this work is simplified. The focus is solely on the vehicular dynamics based only on the front wheels braking.

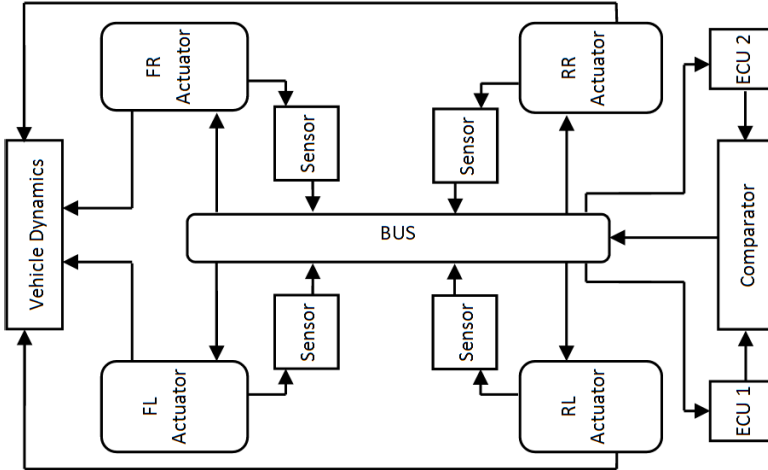


Fig. 4 An automotive brake-by-wire system

The failure behaviour of the actuators coupled with its failure data are modelled below. ‘IF’ represents the Internal Failure of a component, ‘V’ represents a value deviation (i.e., an error in a signal), and ‘C’ represents inadvertent commission of the component. {X} means that a failure can refer to any of FR (front-right), FL (front-left), RL (rear-left), or RR (rear-right), e.g. C_ActuatorFL is the front-left braking actuator. “+”, “.” and “!” are for logical OR, AND and POR respectively.

$$\begin{aligned}
 C_Actuator\{X\} &= IF_Actuator\{X\} + C_BusCommand\{X\} \\
 C_BusCommand\{X\} &= C_Comparator\{X\} + IF_Bus \\
 C_Comparator\{X\} &= C_ECU1\{X\} . C_ECU2\{X\} + IF_Comparator \\
 C_ECU\{X\} &= V_BusSignal\{X\} + IF_ECU \\
 V_BusSignal\{X\} &= V_SensorData\{X\} \\
 V_SensorData\{X\} &= IF_Sensor\{X\}
 \end{aligned}$$

A thorough qualitative analysis on the entire BBW system produces approximately 75 MCSQs. A full quantification of the entire MCSQs is outside the scope of this paper. However, to demonstrate the nearly simultaneous scenario explained in this work we focus on the effects of inadvertent commission of the front right and left actuators.

The vehicle dynamics section provides the overall failures of the system. In this work, we focus only on the front two wheels, in which case we have three possible scenarios:

- The left wheel brakes either alone or before the right wheel, causing the vehicle to veer suddenly to the left.
- The right wheel brakes either alone or before the left wheel, causing the vehicle to veer to the right.
- Both wheels brake at approximately the same time.

How critical these failures are depends on whether the vehicle is in a left-hand or right-hand drive country, but for the sake of the example let us assume it is left-hand drive (i.e., vehicles drive on the left). In this case, veering to the right may cause the vehicle to veer into oncoming traffic, which is the most severe possible failure. Veering to the left may cause the vehicle to go off road, which is still dangerous but less so than a head-on collision with another vehicle. Finally, if both wheels brake at the same time, the car is likely to brake in a roughly straight line, which is the least severe of the three types of failure.

These failures can be represented using temporal gates like so:

$$\text{VeerIntoOncomingTraffic} = C_Actuator_FR \mid C_Actuator_FL$$

$$\text{VeerOffRoad} = C_Actuator_FL \mid C_Actuator_FR$$

$$\text{StraightBraking} = C_Actuator_FL \ \&_d \ C_Actuator_FR$$

Note that here we use the pSAND for the straight-line braking; as long as the brakes fail within an interval of about 0.1 seconds, the result will be the less severe straight braking rather than a dangerous veer to the side. The actuators need not fail at exactly the same moment.

Table 1 Failure Probabilities

Component	Failure rate/hr
C_Actuator_FL	1E-3
C_Actuator_FR	1E-3

Table 2 Results of StraightBraking ($C_Actuator_FL \ \&_{0.1} \ C_Actuator_FR$)

Time (hrs)	Analytical Solution	Monte Carlo Solution	Percentage Error
1E1	5.0194E-5	4.9000E-5	2.37
1E2	4.3873E-4	4.2600E-4	2.90
1E3	1.1849E-3	1.2110E-3	2.21

5 Discussion and Evaluation

Table 2 contains results of the application of both analytical and Monte Carlo solution described in this work on the StraightBraking failure described in the case study. Results for both techniques were achieved by modelling them in Mathematica 8; an application for performing many kinds of computations [24]. By using small realistic failure data with no dynamic stopping techniques or importance measures, we use a large number of trials (10^6) to increase the accuracy of results. MC simulation is also simplified in this work. It is used just for the purpose of demonstration.

From Table 2, it is clear that both techniques produce results close to each other at least to the magnitude. The percentage error is less than 3% which can be considered to be good considering the fact that we have not used a dynamic stopping technique on the Monte Carlo approximation technique.

In general, the results show that the probability of both C_Actuator_FR and C_Actuator_FL increases with time. Meaning the more time given, the more likely it is for the two events to occur within d .

From equation (9), it can be seen that if d (which is equal to $t_1 - t_0$) is zero, the pSAND probability is also zero which means it technically becomes a SAND.

Although we have provided an analytical solution in this work, we have also provided a simulative alternative. This is not to necessarily evaluate the analytical solution but to provide a framework for estimating the pSAND probability of events with distributions other than an exponential distribution.

For future work, one may consider the possibility of parameterising the PAND and POR gates to achieve some form of completion [5] [9].

6 Conclusion

In modern dynamic systems, it is often necessary to be able to represent scenarios involving events which are "nearly simultaneous", i.e., they occur together within a short period of time. It is common for the effects of such failures to be different if they occur nearly simultaneously compared to occurring further apart, as in the braking system presented earlier. However, it is difficult to model this situation in current FTA approaches. Pandora, a modification to FTA, provides the SAND gate to represent simultaneous occurrence of events, but does not cater for any delay between occurrences. This paper extends Pandora's definition of SAND gates to provide "parameterised SAND" gates or pSANDs, which can be used in modelling and evaluating nearly simultaneous scenarios. We have proposed new logical representations for pSAND and provided its exact and approximated calculations using analytical and Monte Carlo solutions respectively. The result is a more flexible analytical approach that enables Pandora-based FTA to be applied to a greater variety of situations in modern safety critical systems.

References

- [1] Gulati, R., Dugan, J.B.: A modular approach for analyzing static and dynamic fault trees. In: Reliability and Maintainability Symposium (1997)
- [2] Merle, G., Roussel, J.: Algebraic modelling of Fault Trees with Priority AND gates. In: 1st IFAC Workshop, pp. 175–180 (2007)
- [3] Dugan, J.B., Doyle, S.A.: New results in fault-tree analysis. In: Tutorial Notes of the Annual Reliability and Maintainability Symposium (1997)
- [4] Dugan, J.B., Bavuso, S.J., Boyd, M.A.: Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Transactions on Reliability* 41(3), 363–377 (1992)
- [5] Walker, M., Papadopoulos, Y.: Synthesis and analysis of temporal fault trees with PANDORA 2: The time of Priority AND gates. *Nonlinear Analysis: Hybrid Systems* 2(2), 368–382 (2008)
- [6] Fussell, J.B., Aber, E.F., Rahl, R.G.: On the quantitative analysis of Priority-AND failure logic. *IEEE Transactions on Reliability* R-25(5), 324–326 (1976)
- [7] Walker, M., Papadopoulos, Y.: Pandora 2: The time of priority-OR gates. In: IFAC Workshop on Dependable Control of Discrete Event Systems (2007)
- [8] Walker, M.: Pandora: a logic for the qualitative analysis of temporal fault trees. Dissertation, University of Hull (2009)
- [9] Vesely, W.E., Stamatelatos, M., et al.: Fault tree handbook with aerospace applications. NASA Office of Safety and Mission Assurance, Washington DC (2002)
- [10] Durga, R., Gopika, V., et al.: Dynamic fault tree analysis using Monte Carlo simulation in probabilistic safety assessment. *Reliability Engineering & System Safety* 94(4), 872–883 (2009)
- [11] Yuge, T., Yanagi, S.: Quantitative analysis of a fault tree with Priority AND gates. *Reliability Engineering & System Safety* 93(11), 1577–1583 (2008)
- [12] Long, W., Sato, Y., Horigome, M.: Quantification of sequential failure logic for fault tree analysis. *Reliability Engineering & System Safety* 67(3), 269–274 (2000)
- [13] Edifor, E., Walker, M., Gordon, N.: Quantification of priority-OR gates in temporal fault trees. In: Ortmeier, F., Lipaczewski, M. (eds.) SAFECOMP 2012. LNCS, vol. 7612, pp. 99–110. Springer, Heidelberg (2012)
- [14] Merle, G., Roussel, J.: Probabilistic algebraic analysis of fault trees with priority dynamic gates and repeated events. *IEEE Transactions on Reliability* 59(1), 250–261 (2010)
- [15] Chaochen, Z., Hoare, C., Ravn, A.: A calculus of Durations. *Information Processing Letters* 40(5), 269–276 (1991)
- [16] Hansen, K.M., Anders, P.R., Stavridou, V.: From safety analysis to software requirements. *IEEE Transactions on Software Engineering* 24(7), 573–584 (1998)
- [17] Palshikar, G.: Temporal fault trees. *Information and Software Technology* 44(3), 137–150 (2002)
- [18] Gorski, J., Wardzinski, A.: Deriving real-time requirements for software from safety analysis. In: Real-Time Systems, pp. 9–14 (1996)
- [19] Schellhorn, G., Thums, A., Reif, W.: Formal fault tree semantics. In: Proceedings of The Sixth World Conference on Integrated Design & Process Technology (2002)
- [20] Gudemann, M., Ortmeier, F., Reif, W.: Computation of Ordered Minimal Critical Sets. In: Proceedings of the 7th Symposium on Formal Methods for Automation and Safety in Railway and Automotives (2008)

- [21] Babczyński, T., Lukowicz, M., Magott, J.: Time coordination of distance protections using probabilistic fault trees with time dependencies. *IEEE Transactions on Power Delivery* 25(3), 1402–1409 (2010)
- [22] Rocco, C., Muselli, M.: A machine learning algorithm to estimate minimal cut and path sets from a Monte Carlo simulation. In: *Probabilistic Safety Assessment and Management (PSAM7–ESREL)*, pp. 3142–3147 (2004)
- [23] Chan, J.C., Kroese, D.P.: Rare-event probability estimation with conditional Monte Carlo. *Annals of Operations Research* 189(1), 43–61 (2009)
- [24] Wolfram Research, *What Is Mathematica?* (2013), <http://www.wolfram.co.uk/mathematica> (accessed January 07, 2013)

Improving of Non-Interactive Zero-Knowledge Arguments Using Oblivious Transfer

Alexander Frolov

National Research University Moscow Power Engineering Institute,
111250. Moscow, Krasnokazarmennaya, Russian Federation
abfrolov@gmail.com

Abstract. We study non-interactive zero-knowledge (NIZK) arguments using oblivious transfer (OT) that correspond to interactive proof protocols but assuming that the prover is computationally bounded. As opposed to the single theorem NIZK proof protocols using common random string, NIZK argument protocols using OT are «multilingual» that is language L or the one-way function can be chosen and declared by prover in non-interactive mode. These protocols use m -out-of- n OT with public keys given by verifier to prover in the initialization phase and common element with unknown to prover and verifier pre-image. It is shown that due to usage of different verifier's secret encryption keys the implementation of NIZK argument protocols can be simplified using a single randomizer for p successive elementary transactions. For systems using 1-out-of-2 OT, proposal allows increase the information rate approximately to $5p/(3p+1)$ times or reduce the soundness probability of NIZK arguments to the same degree. The above factor for single use NIZK is about two that corresponds to almost quadratic decreasing of soundness probability. For NIZK argument using $t+1$ -out-of- 2^t OT ($t > 1$), it is shown that its soundness probability for small t is essentially lower in comparison with soundness probability of NIZK arguments using 1-out-of-2 OT.

1 Introduction

The interactive and non-interactive zero-knowledge protocols are the most important cryptographic primitives of contemporary cryptosystems such as electronic payment systems, electronic voting systems, privacy preserving smart metering systems, etc. They provide the identification of cryptographic protocols participants. The zero-knowledge protocol $(P, V)(x)$ is performed by a prover P and a verifier V possessing a common information x that may be the value $z=f(s)$ of one-way function $f(x)$ which pre-image s is the secret of the prover. Involving the protocol, the prover P convinces the verifier that it possesses the secret s . These protocols have two probabilistic characteristics: *completeness probability* σ (the lower bound of a probability of successive proving for honest prover) and *soundness probability* δ (the upper bound of a probability of successive proving for dishonest prover \tilde{P} that does not possess the secret s). A lowering of this

threshold means improving of protocol soundness. In this chapter, we consider the protocol with completeness probability $\sigma=1$ and soundness probability $\delta \leq \frac{1}{2}$. If the elements with secret pre-images s can be considered as elements of language L corresponding to known witness w , the prover P can convince the verifier V that $z \in L$ with probability at least σ . In this case the soundness probability is the upper bound of the probability of successive proving by dishonest prover that an element $\tilde{z} \notin L$ belongs to L and protocol $(P, V)(x)$ is a protocol for language L . The third characteristic is the *perfectness* that is complete hiding of secret during the protocol performance. The *information rate* depends on the length of transaction that prover sends to verifier, it is the more the shorter transaction.

There are two types of zero-knowledge protocols: interactive and non-interactive protocols. An interactive protocol as usual possesses tree rounds:

- 1) A message *commit* that is the value c of one-way function corresponding to current randomly chosen secret value *committal* of its argument is sent from prover to verifier.
- 2) A message *challenger* that is randomly chosen binary string e of length t , $t \geq 1$, is sent from verifier to prover.
- 3) The message *response* that is the value r depending on committal, challenger and the secret s (the latter is hidid by random committal) is sent from prover to verifier.

After these rounds the verifier verifies the response computing the value of predicate $\text{Verify}(c, e, r, z)$. If it gets the value *true* it accepts otherwise rejects.

These protocols with soundness probability δ can be executed p times. If each time the result of verification is *true*, the verifier accepts with soundness probability δ^p , otherwise rejects.

Interactivity is very inconvenient property of these protocols because it requires direct contact of prover and verifier and is associated with time spent in communication.

In contrast to interactive protocols, the non-interactive protocols use pre-obtained information and protocols are performed without verifier's challengers. These are protocols with common random string. In addition there are two approaches to transformation of interactive protocols in non-interactive protocols:

- transformation using Fiat–Shamir heuristic;
- transformation using oblivious transfer.

Non-interactive protocols with common random string as well as the transformations of the first type are investigated in many issues whereas protocols using oblivious transfer are considered less. In these protocols a non-interactive communication phase is preceded by an initialization phase oblivious transfer parameters initialization. Use of the oblivious transfer makes it necessary to constrain the prover computational capabilities. It follows that using oblivious transfer non-interactive analogue of an interactive zero-knowledge proof protocol has to be considered as an argument protocol. In spite of these shortcomings,

protocols of this type have no restrictions on the number of theorems proved (argumented) for given language and, moreover, are "multilingual" in the sense that in the communication phase based on the once initialized oblivious transfer parameters there are possible arguments for different languages, or one-way functions.

In this chapter, we compare the soundness probabilities of protocols running approximately the same time i.e. possessing approximately equal information rates and we compare the information rates of protocols with approximately the same soundness probabilities.

The rest of this chapter is organized as follows. Section 2 is devoted to related works and announces our proposals. In section 3 the traditional non-interactive analogues of interactive zero-knowledge protocols with binary challengers and proposed their improved variants are discussed, the soundness probabilities are compared and the effectiveness of novel protocols is estimated. In section 4 novel non-interactive analogues of interactive protocols with multibit challengers and their improved variants are introduced with comparison of their soundness and effectiveness w.r.t. protocols considered in previous section. In section 5 the particularity of zero-knowledge arguments for languages is discussed and one example of non-interactive zero-knowledge argument for language is given to illustrate it. Section 6 contains conclusion remarks and description of the open problem.

2 Related Works and Our Proposals

Interactive zero-knowledge protocols has been introduced and studied by S. Goldwasser, S. Micali, and C. Rakoff [1]. The notion of a non-interactive zero-knowledge (NIZK) proof system using common random string has been introduced by M. Blum et al. [2,3,4,5]. The problem with this NIZK was that one proof completely used up the common random string, and to produce more proofs one need to refresh it. Blum et al. [4] showed a single-prover multi-theorem NIZK proof system for 3SAT, and since 3SAT is NP-complete, the result followed for any language in NP [6]. M. Chase and A. Lysyanskaya [6] proposed a construction of a simulatable verifiable random function for achieving multi-theorem NIZK for any language L without having to reduce instances of L to instances of any NP-complete languages as it is suggested [7,8].

The security of Fiat-Shamir heuristic proposed in [9] is questionable [10].

The idea of non-interactive zero-knowledge protocol using 1-out-of-2 oblivious transfer has been represented by N. Kobitz in [11]. It is shortly described in the next section.

The notion of oblivious transfer (OT) has been introduced by M. Rabin [12] to help solve the problem of "exchanging secrets," studied by M. Blum [13]. So the simplest case of OT is the transmission of one bit with the probability $1/2$.

In [14] using ElGamal encryption [15], the 1-out-of-2 OT protocol ($OT_{\frac{1}{2}}$ protocol) is established. In this protocol one party A can transfer two messages to the other party B in such a way that:

- B receives only one message of your choice,
- A does not know which of two messages B received.

The next most essential step was made by M. Bellare and R.L. Rivest [16]. They developed the theory of fractional oblivious transfer and proposed the interactive and non-interactive m/n fractional OT protocols in which one party A can transfer a message to the other party B in such a way that:

- B receives the message with the probability m/n ,
- A is oblivious as to whether the transfer was successful or not, that is A does not know whether B received the message.

In [17] the basic notion of $OT_{\frac{1}{2}}$ protocol was extended to 1-out-of- n OT protocol (OT_n^1 protocol).

The Bellare–Rivest $1/n$ fractional OT and $(n-1)/n$ fractional OT protocols generalize the $1/2$ fractional OT protocols, but their other m/n OT protocols ($1 < m < n-1$) are based on the original polynomial scheme. All these protocols involve n ElGamal encryptions. In [18] this scheme is applied to calculate the public keys of m -out-of- n oblivious transfer protocol (OT_n^m protocol) with repeated use of ElGamal. In such protocols, one party A can transfer n messages to the other party B in such a way that:

- B receives combination of m messages of your choice,
- A does not know which of possible messages combinations B received.

In [19] on the basis of other matrix scheme, the interactive and non-interactive OT_n^m protocols which repeatedly use the Nyberg–Rueppel digital signature with message recovery [20] have been described.

The OT protocols considered in [21] start with the *Global set-up phase* when the group G of large prime order q with intractable the discrete logarithm and the Diffie–Hellman problems, its generator b , its element U with unknown for both parties discrete logarithm $\log_b U$ are declared.

The non-interactive OT protocols [21] implemented in this chapter involve the *Receiver’s public key set-up and certification phase* when receiver’s public key is set-upped and sent to trusted center for verifying and publishing for the prover.

During the communication phase of non-interactive m -out-of- n oblivious transfer protocol (NIOT $_n^m$ protocol), n ElGamal encryptions are involved repeatedly. In each case the encryption or signature use a new randomizer. Within a single session of OT protocol, encryptions or signatures carried out repeatedly using principally different secret keys. Therefore, as is proved in [21], the re-use

of randomizer is safe. As a consequence, the complexity of sender's calculation of randomizers is reduced by n times, the amount of transmitted information as well as the number of exponentiation is reduced which makes the protocol more efficient. This effective version of NIOT_n^m protocol is denoted NIEOT_n^m [21].

Implementing of NIEOT_n^m allows increasing the information rate of zero-knowledge arguments preserving the soundness or decreasing the soundness probability at the same information rate. Moreover we show that the randomizer reusing in distinct p runs of non-interactive analogues of zero-knowledge proofs with binary challengers ($\text{NIZKOT}_2^1(p)$) is safe. This allows us to introduce the effective variants $\text{NIZKEOT}_2^1(p)$ of these analogues and estimate their soundness probability and effectiveness. We prove that NIZKEOT_2^1 are secure in standard model.

We introduce $\text{NIZKOT}_{2^t}^{t+1}$ argument as non-interactive analogue of interactive zero-knowledge proofs or arguments with multibit challengers (for example Schnorr interactive protocol). Implementing of $\text{NIEOT}_{2^t}^{t+1}$ and reusing of randomizer in distinct runs of protocol allows us propose and estimate the effective version $\text{NIZKEOT}_{2^t}^{t+1}$ of that protocol. We estimate the effectiveness of

NIZKEOT_n^m by the fraction $\rho_{\text{NIZKEOT}_n^m} = \frac{e_{\text{NIZKOT}_n^m}}{e_{\text{NIZKEOT}_n^m}}$ expressed the increase in the information rate of NIZKEOT_n^m w.r.t. NIZKOT_n^m . The nominator and denominator of this fraction represent the length of NIZKOT_n^m (NIZKEOT_n^m) transactions.

3 Comparison of NIZKOT_2^1 and NIZKEOT_2^1 Arguments

In this section we represent NIZKOT_2^1 arguments corresponding to interactive zero-knowledge arguments with binary challengers.

Accordingly to Koblitz idea [11], it is supposed that the prover P at the initialization phase received the long sequence of verifier's public keys (β_i, β_{2i}) , $i=1, \dots, p$ for p runs of 1-out-of-2 oblivious transfer. This sequence can be used by prover in many distinct zero-knowledge arguments. The prover P imitating the logic of interactive protocol with binary challengers, in each of p iterations sends the verifier in non-interactive mode the commit c_i and then sends obliviously two

responses (r_{0_j}, r_{1_j}) to both possible binary challengers 0 and 1 correspondingly.

The prover reads of your choice one of them. The prover does not learn what to choose verifier. Then verifier calculates the value of predicate as it does this in interactive protocol. The soundness probability of this protocol is equal to 2^p . As a result, an effect of the interactive protocols is reached in a non-interactive mode. To implement this idea, one uses the probabilistic encryption, for example ElGamal encryption [15]. In [11] it is proposed to implement additive hiding instead of multiplicative: the second message $C_2=m\beta^y$ of ElGamal cryptogram where m is the message, β is ElGamal public key and y is a randomizer is replaced by $C_2=m\oplus\psi(\beta^y)$. Here $\psi:G\rightarrow\{0,1\}^n$ is an invertible mapping (G is the basic group of ElGamal cryptosystem). Using the generator α , the secret key x , and the first message $C_1=\alpha^y$ of the ElGamal cryptogram, the decryption is involved as $C_2\oplus\psi(C_1^y)=C_2\oplus\psi(\alpha^{xy})=C_2\oplus\psi(\beta^y)=m$. In [21], it is shown that due to using of different secret keys in two or more encryptions the reusing of randomizer is safe. In this case, the information rate of communication phase of OT protocols increases. In this chapter the idea of randomizer reusing is extended to sequentially executed runs of non-interactive zero-knowledge arguments. As a result, the soundness probability essentially decreases. In this section the decreasing degree will be estimated.

Let us NIZKOT 1_2 argument protocol runs p times with different commits and different ElGamal encryption keys. The protocol uses NIOT 1_2 organized using the large multiplicative group G with generator α , the element U which discrete logarithm is unknown both prover and verifier. Suppose that these parameters are installed on the initialization phase. Additionally, on that phase the verifier V chooses the sequence of secret keys: (i_j, x_j) , $j=1, \dots, t$, where $i_j=e_j+1 \in \{1,2\}$ correspond to binary challengers $e_j \in (0,1)$, $x_j \in \{2, \dots, \text{ord } \alpha - 1\}$ are the secret keys of ElGamal cryptograms, calculates and sends to trusted center the sequence of public keys:

$$((\beta_{11}, \beta_{21}), \dots, (\beta_{1j}, \beta_{2j}), \dots, (\beta_{1t}, \beta_{2t})), \beta_{ij} = \alpha^{x_j i_j}, \beta_{3-i_j} = U \alpha^{-x_j}, \quad (1)$$

$$j=1, \dots, p.$$

The trusted center publishes it after verification that for all j the equalities $\beta_{1j}\beta_{2j}=U$ are valid. The *prover* gets this sequence from trusted center. Description of initialization phase above is given in multiplicative notion. It can be rewritten in an additive mode implementing the group of an elliptic curve points.

Moreover, using of the Koblitz additive hiding mode allows separate the algebraic structures using for OT transfer and zero-knowledge arguments on the

merits. For example, OT based on the multiplicative group can be used in the implementation of zero-knowledge argument of discrete logarithm of the elliptic curve point.

Now we can consider the general scheme of communication, non-interactive phase in two variants: traditional and accelerated supposing that we would like to get a non-interactive version of known interactive protocol which runs consist of the commit $c_j = f(l_j)$ of randomly chosen committal l_j (c_j is sent from prover to verifier), randomly chosen binary challengers $e_j \in \{0,1\}$ (from verifier to prover), the response $r_j \in \{r_{0j}, r_{1j}\}$, $r_j = r_{e_j} = l_j \circ (s \bullet e_j)$ (from prover to verifier) and the verification steps $\text{Verify}(c_j, e_j, r_j, z)$ executed by the verifier. Above \circ is a multiplication or an addition, \bullet is a rising to power or multiplication depending on type of function f . (In this section we consider NIZK arguments of knowledge a pre-image of one-way function, the NIZK arguments peculiarities for languages we left to fifth section).

Consider a traditional variant: protocol $\text{NIZKOT}_{\frac{1}{2}}(p)$.

The prover randomly chooses p committals l_j and calculates the sequence c of p commits $c_j = f(l_j)$, $j=1, \dots, p$. Then it calculates the current parameters for OT transactions. This is the sequence

$$y = \left((y_{11}, y_{21}), \dots, (y_{1j}, y_{2j}), \dots, (y_{1p}, y_{2p}) \right)$$

of randomly chosen pairs of distinct secret randomizers. Using those randomizers and verifier's public keys (1), the prover calculates the pairs of possible responses (m_{1j}, m_{2j}) , $m_{ij} = r_{i-1,j}$, $i=1, 2$, $j=1, \dots, p$ and the sequence of OT transactions

$$\text{OT}(m_{11}, m_{21}), \dots, \text{OT}(m_{1j}, m_{2j}), \dots, \text{OT}(m_{1p}, m_{2p})$$

where

$$\text{OT}(m_{1j}, m_{2j}) = \left((\alpha^{y_{1j}}, m_{1j} \oplus \psi(\beta_{1j}^{y_{1j}})), (\alpha^{y_{2j}}, m_{2j} \oplus \psi(\beta_{2j}^{y_{2j}})) \right). \quad (2)$$

Finally, the prover sends to the verifier the sequence c of commits and the sequence (2) of OT transactions.

From the triplets $(c_j, (\alpha^{y_{1j}}, \alpha^{y_{2j}}), ((m_{1j} \oplus \psi(\beta_{1j}^{y_{1j}})), (m_{2j} \oplus \psi(\beta_{2j}^{y_{2j}}))))$, the prover gets accordingly to its secret keys (i_j, x_j) the set of messages

$$(m_{i_1}, \dots, m_{i_j}, \dots, m_{i_t})$$

obtaining m_{i_j} as following:

$$(m_{ij} \oplus \psi(\beta_{ij}^{y_{ij}})) \oplus \psi(\alpha^{y_{ij}x_j}) = m_{ij} \oplus \psi(\beta_{ij}^{y_{ij}}) \oplus \psi(\beta_{ij}^{y_{ij}}) = m_{ij} = r_{i-1,j} = r_{e_j} = r_j.$$

The verifier verifies these p messages calculating the values of predicates $\text{Verify}(c_j, e_j, r_j, z)$. If at least one of them is false, the verifier rejects, otherwise it accepts the argument.

Example 3.1. The zero-knowledge argument of knowledge discrete logarithm of the element $z=b^s$ to the base b . Here $f(x)=b^x$;

$$c_j = b^{l_j}; r_{0j} = l_j; r_{1j} = l_j + s; \text{Verify}(c_j, e_j, r_j, z) : b^{r_j} = c_j z^{e_j}.$$

Example 3.2. The zero-knowledge argument of knowledge of the square root modulo composite n . Here $f(x)=x^2 \bmod n$;

$$c_j = l_j^2; r_{0j} = l_j s^0; r_{1j} = l_j s^1; \text{Verify}(c_j, e_j, r_j, z) : r_j^2 = c_j z^{e_j}.$$

Remark that both examples could be implemented on the same oblivious transfer parameters initialized only ones.

The accelerated variant of zero-knowledge argument protocol $\text{NIZKEOT}_{\frac{1}{2}}(p)$ differs from just considered traditional $\text{NIZKOT}_{\frac{1}{2}}(p)$ protocol such that instead of the sequence \mathbf{y} there is randomly chosen a unique randomizer y . Instead of OT transactions (2) there are calculated and transposed the element α^y and OT transactions

$$\text{OT}(m_{1j}, m_{2j}) = (m_{1j} \oplus \psi(\beta_{1j}^y), m_{2j} \oplus \psi(\beta_{2j}^y)), j=1, \dots, p.$$

Using the element α^y these transactions can be represented as a sequence of cryptogram pairs

$$\text{OT}(m_{1j}, m_{2j}) = ((\alpha^y, m_{1j} \oplus \psi(\beta_{1j}^y)), (\alpha^y, m_{2j} \oplus \psi(\beta_{2j}^y))), j=1, \dots, p.$$

Finally, the prover sends to the verifier both the sequences c of commits and the sequence of OT transactions.

The set of messages

$$(m_{i_1}, \dots, m_{i_j}, \dots, m_{i_t})$$

is getting by verifier accordingly to its secret keys from the triplets $(c_j, \alpha^y, (m_{1j} \oplus \psi(\beta_{1j}^y), m_{2j} \oplus \psi(\beta_{2j}^y)))$, obtaining m_{i_j} as following:

$$(m_{ij} \oplus \psi(\beta_{ij}^y)) \oplus \psi(\alpha^{yx_j}) = (m_{ij} \oplus \psi(\beta_{ij}^y)) \oplus \psi(\beta_{ij}^y) = m_{ij} = r_{i-1,j} = e_{e_j} = r_j.$$

Let us compare the soundness probabilities that can be guaranteed by these protocols running approximately the same time. During p runs of traditional protocol on the communication non-interactive phase there are transferred $5p$ elements of group G . To calculate them the prover executed the same number of exponentiations. In the accelerated protocol this number is decreased up to $3p+1$.

So the protocols $\text{NIZKOT}_{\frac{1}{2}}(3)$ and $\text{NIZKEOT}_{\frac{1}{2}}(5)$ run approximately the same time with equal information rate providing the soundness probabilities

$$\delta_{\text{NIZKOT}_{\frac{1}{2}}(3)} = \left(\delta_{\text{NIZKOT}_{\frac{1}{2}}} \right)^3 \text{ and } \delta_{\text{NIZKEOT}_{\frac{1}{2}}(5)} = \left(\delta_{\text{NIZKEOT}_{\frac{1}{2}}} \right)^5.$$

Taking into account that $\delta_{\text{NIZKOT}_{\frac{1}{2}}} = \delta_{\text{NIZKEOT}_{\frac{1}{2}}} = \frac{1}{2}$, one can see that $\left(\delta_{\text{NIZKOT}_{\frac{1}{2}}(3p)} \right)^5 = \left(\delta_{\text{NIZKOT}_{\frac{1}{2}}} \right)^{15p} \approx \left(\delta_{\text{NIZKEOT}_{\frac{1}{2}}(5p)} \right)^3 = \left(\delta_{\text{NIZKEOT}_{\frac{1}{2}}} \right)^{15p}$

It follows that

$$\delta_{\text{NIZKEOT}_{\frac{1}{2}}(5p)} \approx \left(\delta_{\text{NIZKOT}_{\frac{1}{2}}(3p)} \right)^{\frac{5}{3}}. \quad (3)$$

So, with equal information rate, implementing of $\text{NIZKEOT}_{\frac{1}{2}}$ instead of $\text{NIZKOT}_{\frac{1}{2}}$ entails essential improving of soundness. On the other hand, $e_{\text{NIZKEOT}_{\frac{1}{2}}(15p)} \approx 45p$ and $e_{\text{NIZKOT}_{\frac{1}{2}}(15p)} \approx 75p$.

Hence providing the same soundness implementing of $\text{NIZKEOT}_{\frac{1}{2}}$ instead of $\text{NIZKOT}_{\frac{1}{2}}$ entails essential increasing of information rate:

$$\rho_{\text{NIZKEOT}_{\frac{1}{2}}(5p)} \approx \frac{5}{3}. \quad (4)$$

Let us denote $\text{SUNIZKOT}_{\frac{1}{2}}$ and $\text{SUNIZKEOT}_{\frac{1}{2}}$ the single use protocols $\text{NIZKOT}_{\frac{1}{2}}(p)$ and $\text{NIZKEOT}_{\frac{1}{2}}(p)$. In these protocols the sequence c of commits is generated and transferred to verifier in the initialization phase and in the communication phase there are sent $2p+1$ elements. As a result, during of $2p$ transactions $\text{SUNIZKOT}_{\frac{1}{2}}$, $4p+1$ transactions $\text{SUNIZKEOT}_{\frac{1}{2}}$ can be performed. Hence the protocols $\text{SUNIZKOT}_{\frac{1}{2}}(2)$ and $\text{SUNIZKEOT}_{\frac{1}{2}}(4)$ run approximately the same time. It follows that

$$\delta_{\text{SUNIZKEOT}_{\frac{1}{2}}(4p)} \approx \left(\delta_{\text{SUNIZKOT}_{\frac{1}{2}}(2p)} \right)^2, \rho_{\text{SUNIZKEOT}_{\frac{1}{2}}(4p)} \approx 2. \quad (5)$$

So, the reusing of randomizers entails essential decreasing of soundness probability (almost squaring for single use protocols).

Now let us recall the proof of randomizer reusing safety in $\text{NIOT}_{\frac{1}{2}}$ [21] implemented in $\text{NIZKOT}_{\frac{1}{2}}$ taking into account additive hiding mode.

Claim 1. *The problem of the second message extraction reusing the randomizer of NIOT 1_2 transactions and the Diffie–Hellman problem are polynomially equivalent.*

Proof. Let us from the triplet $(c_j, \alpha^y, (m_{1j} \oplus \psi(\beta_{1j}^y), m_{2j} \oplus \psi(\beta_{2j}^y))) = (c_j, \alpha^y, (C_{1j}, C_{2j}))$ and the secret key (i_j, x_j) of the verifier the message $m_{i_j} = C_1 \oplus \psi(\beta_{i_j}^y)$ is calculated. For calculation of the second message $m_{(3-i)_j} = C_2 \oplus \psi(\beta_{(3-i)_j}^y)$, one have to find $\beta_{3-i_j}^y$ that is it is necessary to solve the Diffie–Hellman problem: from known values $\alpha, \alpha^y, \beta_{3-i_j} = \alpha^{x'}$ one must find $\beta_{3-i_j}^y = \alpha^{yx'}$. The value $\beta_{i_j}^y$ that allows computing the value m_{i_j} is useless for computing of the value $\beta_{3-i_j}^y$, required for computing of the second message m_{3-i_j} :

$$\beta_{3-i_j}^y = U^y (\beta_{i_j}^y)^{-1}, \quad (6)$$

because the prover have to know the secret randomizer y known only to verifier. Let us under known α and α^y the message m_{3-i_j} and then the message $\beta_{3-i_j} = \psi^{-1}(C_2 \oplus m_{3-i_j})$ have been computed using an effective algorithm. Let us put $U = \alpha^z$. Then from equation (6) we compute $U^y = (\beta_{i_j}^y)(\beta_{3-i_j}^y)^{-1}$. Hence if we can compute m_{3-i_j} , we can solve the Diffie–Hellman problem: using known $\alpha, \alpha^z, \alpha^y$ one can computer α^{zy} . So the following claim has been proved.

Corollary 1. *The reusing of randomizer within one iteration NIZKEOT 1_2 argument is safe.*

One can remark that the reusing of randomizer y in the distinct runs of NIZKEOT $^1_2(p)$ arguments the more safely because in contrast to the keys used within one iteration verifiers secret keys used in various iterations are algebraically disconnected.

Corollary 2. *NIZKEOT 1_2 argument is secure in standard model.*

Corollary 3. *The reusing of randomizer y in all runs of NIZKEOT $^1_2(p)$ is safe.*

Corollary 4. *The approximations and estimations (3-5) are valid.*

4 NIZKOT_{2^t}^{t+1} and NIZKEOT_{2^t}^{t+1} Arguments (t>1)

In this section we consider non-interactive analogue of interactive zero-knowledge proofs with t -bit challengers (such as for example the Schnorr protocol [22] with challengers from $\{0,1\}^t$), $t > 1$. The problem is that the set of possible responses consists of $n=2^t$ elements. Implementing of NIZKOT_n¹ protocol is less effective: in this case the soundness probability equals 2^{-t} that can be provided by NIZKEOT₂¹(t) possessing much greater information rate.

On the initialization phase of NIZKOT_n^m and NIZKEOT_n^m, a verifier V and a prover P generate the system parameters of NIOT_n^m (the group G of large prime order q , its generator g and an element U with unknown to both the users discrete logarithm $\log_g U$), the verifier V randomly chooses the secret keys, calculates and publishes corresponding public keys allowing it to receive obviously to prover m out of n messages. Details of the initialization phase we put to the end of section.

Consider the non-interactive communication phase. Let us $n = 2^t, m=t+1$ and P has to argue the verifier V in possessing the pre-image s of the declared value $z=f(s)$ of one-way function $f(x)$ using NIZKOT_n^m. The argument consists of the following steps.

Generation and commit: the prover randomly chooses committals l, l_1, \dots, l_t , calculates the commits $c=f(l), c_1=f(l_1), \dots, c_t=f(l_t), \dots, f(l_t)$, and sends $c, c_1, \dots, c_t, \dots, c_t$ to the verifier.

NIOT_n^m: the prover calculates the cortege of $n=2^t$ messages $r_{(e_1, \dots, e_t)} = l + \sum_{i=1}^t e_i l_i + (e_1 \vee \dots \vee e_t) s, e_i \in \{0,1\}$, preserving the lexicographic order of indices (we suppose, that the function $f(x)$ is the function of additive type). Here the values 0 and 1 are interpreted as integer numbers. The prover sends to the verifier obviously to prover m -out-of- n messages using NIOT_n^m.

Verification: the verifier gets m out of n messages $m(e_{i_1}, \dots, e_{i_t}), i=1, \dots, t+1$, corresponding to $t+1$ arbitrary chosen binary t -tuples (e_1, \dots, e_t) unknown to prover and verifies $m=t+1$ predicate values

$$\text{Verify}(c, c_1, \dots, c_t, \dots, c_t, r_{(e_{i_1}, \dots, e_{i_t})}, z):$$

$$f(r_{(e_{i_1}, \dots, e_{i_t})}) = c c_1^{e_{i_1}} \dots c_t^{e_{i_t}} z^{(e_{i_1} \vee \dots \vee e_{i_t})}, i=1, \dots, t+1. \tag{7}$$

If all of them are *true*, the argument can be accepted in opposite case it have to be rejected.

The completeness of this NIZK argument is equal to 1 because if messages were calculated by honest prover, the values of all predicates are *true*.

To estimate the soundness probability, let us remark that dishonest prover \tilde{P} that knows z but oblivious w.r.t. s could calculate at most $t+1$ messages $r_{(e_1, \dots, e_t)}$ satisfying predicate (7). It cannot choose the committal l and calculate the committals l_1, \dots, l_t such that $t+2$ responses $r_{(e_1, \dots, e_t)}$ would be satisfied. If this would be in case, then from $t+2$ equations

$$l + e_{i1}l_1 + \dots + e_{it}l_t + s, \quad i=1, \dots, t+2$$

the value $s = f^{-1}(z)$ that is the pre-image of one way-function would be found.

On the other hand the dishonest prover \tilde{P} can compute $t+1$ messages possessing indices (e_1, \dots, e_t) of weight 0 or 1 and corresponding commits $c, c_1, \dots, c_p, \dots, c_p$ satisfying the predicates as following: choose the committal $l = r_{(0, \dots, 0)}$ randomly and define the commit $c = f(l)$. Then choose randomly t distinct messages $r_{(e_1, \dots, e_t)}$ with indices of weight 1 and calculate the commits $c_i = f(r_i - r)z^{-1}$ were $r_i = r_{(e_1, \dots, e_t)}$ with $e_{i1}=1, i=q, \dots, t, r=l$. Now to calculate the missing message $r_{(e_1, \dots, e_t)}$, one need to know the secret s . Computationally bounded dishonest prover has to choose them randomly. The probability that they will satisfy the predicate is negligible. It follows, that the probability of wrong excepting of $t+1$ out of 2^t messages is at most $\frac{1}{C^{t+1} 2^t}$.

To prove the perfectness, remark that the verifier receives $t+1$ messages that allow getting the system of $t+1$ equations in $t+2$ variables, that has a solution for any fixed value of secret s . That is the security is unconditional.

Now one can estimate the soundness probability in comparison with NIZKOT $_{\frac{1}{2}}$. The transaction of NIZKOT $_{\frac{1}{2}}^{t+1}$ argument is the following:

$$c, c_1, \dots, c_t z \text{ OT}(m_1, \dots, m_{2^t})$$

were OT transaction consists of 2^t ElGamal cryptograms, i.e. it consists of 2^{t+1} group G elements and $t+1$ elements of algebraic structure corresponding to the function f . The soundness probability is estimated as $\frac{1}{C^{t+1} 2^t}$.

If effective (i.e. reusing randomizer) NIEOT $_{\frac{1}{2}}^{t+1}$ is implemented, then the OT transaction consists of one element that is an exponent of reused randomizer and 2^t second elements of ElGamal cryptograms, i.e. transaction consists of $2^t + 1$ group G elements and $t+1$ elements of algebraic structure corresponding to function f . The soundness probability does not change.

NIZKEOT $_{\frac{1}{2}}(2^t)$ transposes $3 \times 2^t + 1$ elements possesses the soundness probability $\left(\frac{1}{2}\right)^{-2^t}$.

Comparing the soundness probabilities, the summands $t+1$ and 1 can be neglected.

It follows that

$$\delta_{\text{NIZKEOT}_{2^t}^{t+1}} \approx \delta_{\text{NIZKOT}_{2^t}^{t+1}(2)} = \left(\delta_{\text{NIZKOT}_{2^t}^{t+1}} \right)^2. \tag{8}$$

During 3 runs of the first protocol with soundness probability

$$\delta_{\text{NIZKOT}_{2^t}^{t+1}(3)} = \left(\frac{1}{C^{t+1}} \right)^3$$

one can perform 2 runs of $\text{NIZKEOT}_{\frac{1}{2}}(2^t)$ with

soundness probability $\delta_{\text{NIZKEOT}_{\frac{1}{2}}(2 \times 2^t)} = \left(\frac{1}{2} \right)^{-2^{t+1}}$.

During 3 runs of the second protocol with the same soundness probability

$$\delta_{\text{NIZKEOT}_{2^t}^{t+1}(3)} = \left(\frac{1}{C^{t+1}} \right)^3$$

only 1 run of $\text{NIZKEOT}_{\frac{1}{2}}(2^t)$ with soundness

probability $\delta_{\text{NIZKEOT}_{\frac{1}{2}}(2^t)} = \left(\frac{1}{2} \right)^{-2^t}$ could be executed. In table 1 there are represented the comparisons of these soundness probabilities.

Table 1 Comparison of soundness probabilities

t	3	4	5	6	7
$\frac{\delta_{\text{NIZKOT}_{\frac{1}{2}}(2^t \times 5)}}{\delta_{\text{NIZKOT}_{2^t}^{t+1}(3)}}$	≈ 100	≈ 5649	< 1	$\ll 1$	$\ll 1$
$\frac{\delta_{\text{NIZKEOT}_{\frac{1}{2}}(2^t \times 2)}}{\delta_{\text{NIZKOT}_{2^t}^{t+1}(3)}}$	$\approx 5,5$	$\approx 19,5$	< 1	$\ll 1$	$\ll 1$
$\frac{\delta_{\text{NIZKEOT}_{\frac{1}{2}}(2^t)}}{\delta_{\text{NIZKEOT}_{2^t}^{t+1}(3)}}$	≈ 1339	$\approx 655 \times 10^3$	$\approx 733 \times 10^5$	$\approx 130 \times 10^5$	$\ll 1$
$\delta_{\text{NIZKEOT}_{2^t}^{t+1}(3)}^{-1}$	$\approx 343 \times 10^3$	$\approx 833 \times 10^8$	$\approx 744 \times 10^{15}$	$\approx 240 \times 10^{24}$	$\approx 292 \times 10^{35}$
$\delta_{\text{NIZKEOT}_{\frac{1}{2}}(2^t)}^{-1}$	256	65536	$\approx 429 \times 10^7$	$\approx 184 \times 10^{15}$	$\approx 340 \times 10^{36}$

From approximation (8) and table 1 one can see that the reusing of randomizer cryptograms entails essential decreasing of soundness probability that made NIZKEOT $_{2^t}^{t+1}$ more effective than NIZKOT $_{2^t}^{t+1}$ and (for $t \leq 6$, providing small soundness probability) much more attractive than NIZKEOT $_{2^t}^1$.

The effectiveness of NIZKEOT $_{2^t}^{t+1}$ argument is estimated as follows:

$$\rho_{\text{NIZKEOT}_{2^t}^{t+1}} = \frac{t+1+2^{t+1}}{t+2+2^t} \approx 2. \tag{9}$$

Terminating this section, we recall from [21] the description and analysis of NIOT $_n^m$ and NIEOT $_n^m$ adapted for Koblitz’s hiding mode.

The NIOT $_n^m$ protocol can be obtained using key-information of the m/n fractional OT protocol [16].

The Global set-up phase of this protocol is completed by letting $\alpha_0=1 \in Z_q$, and fixing of n distinct elements $\alpha_1, \dots, \alpha_n$ of $Z_q \setminus \{\alpha_0\}$. All these elements are public.

The verifier’s key set-up procedure is the following.

The verifier’s public key is a vector $(\beta_1, \beta_2, \dots, \beta_n, W_0, W_1, \dots, W_m) \in G^{n+m+1}$.

To compute it, the verifier V chooses at random a size m subset of $[n]=\{1, 2, \dots, n\}$ specifying an injective map $\pi: [m] \rightarrow [n]$, where $\pi(1), \dots, \pi(m)$ are the m chosen indices. Later he chooses elements $x_{\pi(1)}, \dots, x_{\pi(m)} \in Z_n$ at random and sets $\beta_{\pi(i)} = b^{x_{\pi(i)}} \in G$ for $i=1, \dots, m$. This specifies m elements of $\beta_1, \beta_2, \dots, \beta_n$, in such a way that the verifier V knows their discrete logarithms. The other $n-m$ elements have to be specified in such a way that V doesn’t know and cannot compute their discrete logarithms. To compute these $n-m$ elements, V first compute elements W_0, W_1, \dots, W_m as follows.

The verifier V defines $m+1$ by $m+1$ matrix

$$A = \left(\alpha_{\pi(i)}^j \right), i, j = 0, \dots, m.$$

This is Vandermonde matrix over the field Z_q . It is invertible. V computes its inverse

$$B = A^{-1} = \left(\beta_{ij} \right), i, j = 0, \dots, m.$$

V now sets

$$W_j = U^{\beta_{j,0}} \cdot \prod_i^m \beta_{\pi(i)}^{\beta_{j,i}}, j = 0, \dots, m.$$

Finally, V specifies the remaining elements of the public key:

$$\beta_i = \prod_{j=0}^m W_j^{\alpha_i^j} \text{ for all } i \in [m] \text{ that are not in the range of } \pi.$$

The verification procedure performed by the prover P or the trusted center T is the following:

- 1) check that the public key consists of n elements of G followed by another $m+1$ elements of G ;
- 2) check the predicates

$$U = \prod_{j=0}^m W_j; \beta_i = \prod_{j=0}^m W_j^{\alpha_i^j} \text{ for all } i=1, \dots, n.$$

The communication phase of the NIOT $_n^m$ protocol is the following:

The prover P chooses at random n distinct numbers $0 < y_1, y_2, \dots, y_n < q$, and sends to the verifier V two n -tuples of group G elements

$$(c_1, c_2, \dots, c_n) = (b^{y_1}, b^{y_2}, \dots, b^{y_n})$$

and

$$(\sigma_1, \sigma_2, \dots, \sigma_n) = (m_1 \oplus \psi(\beta_1^{y_1}), m_2 \oplus \psi(\beta_2^{y_2}), \dots, m_n \oplus \psi(\beta_n^{y_n})).$$

This information corresponds to n ElGamal cryptograms

$$(b^{y_i}, m_i \oplus \psi(\beta_i^{y_i})), i=1, \dots, n.$$

For $i=1, \dots, m$, the verifier V calculates

$$\begin{aligned} \sigma_{\pi(i)} c_{\pi(i)}^{-x_{\pi(i)}} &= \sigma_{\pi(i)} b^{-y_{\pi(i)} x_{\pi(i)}} = m_{\pi(i)} \oplus \psi(\beta_{\pi(i)}^{y_{\pi(i)}} b^{-y_{\pi(i)} x_{\pi(i)}}) = \\ &= (m_{\pi(i)} \oplus \psi(\beta_{\pi(i)}^{y_{\pi(i)}})) \oplus \psi(\beta_{\pi(i)}^{y_{\pi(i)}}) = m_{\pi(i)}. \end{aligned}$$

Note that the numbers y_1, y_2, \dots, y_n could be the same and equal to the randomly chosen number y : $y_1 = y_2 = \dots = y_n = y$.

Claim 4.1. *The multiple reuse of randomizer in the NIEOT $_n^m$ protocols is safe.*

Proof. Let the randomizers be the same: $y_1 = y_2 = \dots = y_n = y$. Note that knowing of b^y and $b^{x_i} = \beta_i$, i is not in the range of π , is not sufficient to calculate $\beta_i^y = b^{x_i y}$ since in this case solving of the Diffie–Hellman or the discrete logarithm problems is required. Moreover, the additional knowledge of all messages $m_{\pi(i)}$ does not allow to calculate m_i , since different secret keys $x_{\pi(i)}$ and x_i of ElGamal cryptosystem have been used for the calculation of $\beta_{\pi(i)}$ and β_i . So, the randomizer reuse in separate performances of the NIOT $_n^m$ protocols is safe for the prover: P can be sure that only m messages have been received. At the same time the verifier V , keeping $(x_1, x_2, \dots, x_{m-1}, x_m)$ and π secret, still convinced that the prover P cannot distinguish which of n elements are the elements $\beta_{\pi(i)}$, $i=1, \dots, m$. This completes the proof.

It follows that the communication phase of the NIOT $_n^m$ protocol can be simplified as follows (obtained effective version of NIOT $_n^m$ is denoted NIEOT $_n^m$):

P chooses at random the number $0 < y < q$, calculates and sends to V an element $c = b^y$ and n -tuple of group G elements

$$(\sigma_1, \sigma_2, \dots, \sigma_n) = (m_1 \oplus \psi(\beta_1^y), m_2 \oplus \psi(\beta_2^y), \dots, m_n \oplus \psi(\beta_n^y)).$$

This information corresponds to n ElGamal cryptograms

$$(b^y, m_i \oplus \psi(\beta_i^y)), i=1, \dots, n.$$

For $i=1, \dots, m$, V calculates

$$\begin{aligned} \sigma_{\pi(i)} \oplus \psi(c^{x_{\pi(i)}}) &= \sigma_{\pi(i)} \oplus \psi(b^{yx_{\pi(i)}}) = (m_{\pi(i)} \oplus \psi(\beta_{\pi(i)}^y)) \oplus \psi(b^{yx_{\pi(i)}}) \oplus \psi = \\ &= m_{\pi(i)}. \end{aligned}$$

Corollary 1. *The reusing of randomizer within one iteration NIZKEOT $_n^m$ argument is safe.*

Reusing of randomizer y in the distinct runs of NIZKEOT $_n^m(p)$ arguments is the more safely because in contrast to the keys used within one iteration verifier's secret keys used in various iterations are algebraically disconnected.

Corollary 2. *The reusing of randomizer y in all runs of NIZKEOT $_n^m(p)$ is safe.*

The following question remains open: is it possible to solve the Diffie–Hellman problem using an algorithm for computing of m_i , i is not in the range of π , under the conditions when V knows elements $x_{\pi(i)}$, $m_{\pi(i)} \oplus \psi(\beta_{\pi(i)}^y)$, $\beta_{\pi(i)} = b^{x_{\pi(i)}}$ for $i=1, \dots, m$, $c = b^y$, and U ?

5 Particularity of Arguments for Languages

As it was emphasized in Introduction, implementing oblivious transfer is only possible for computationally bounded prover. It follows that interactive zero-knowledge proof protocols for languages are transformed into non-interactive zero-knowledge argument protocols for languages. These protocols can be implemented using the proposals of previous sections concerning arguments of knowledge the pre-image of one-way functions. Let us consider some examples.

Example 5.1. For language $L = Q_n$ that consists of all quadratic residues (QR) modulo composite n with witness w that is the factorization of the number n , the interactive zero-knowledge proof protocol is the following [23]. The prover arguments the verifier that element z is QR modulo n .

1. The prover P chooses committal l and computes and sends to the verifier V the commit $c = l^2$.
2. The verifier randomly chooses binary challenger $e \in \{0, 1\}$ and sends it to prover.

3. The prover computes the response $r=ls^e$, where s is a square root of z modulo n and sends it to verifier.
4. The verifier computes the predicate $r^2=cz^e$. If it is *true*, the verifier accepts otherwise it rejects.

From the table 2, one can see that the dishonest prover has successive result in argument the verifier that the quadratic non residue \tilde{z} is QR only if it guesses the challenger even it knows the witness. That is the dishonest prover can be computationally unbounded.

Table 2 Situations for dishonest prover

Expected challenger	Commit	Response	Unexpected challenger	Not realizable response
$e=1$	$c=l^2/\tilde{z} \bmod n$	$r=l$	$e=0$	$r=\sqrt{l^2/\tilde{z}} \bmod n$
$e=0$	$c=l^2 \bmod n$	$r=l$	$e=1$	$r=\sqrt{l^2\tilde{z}} \bmod n$

Now consider the non-interactive version of this protocol.

1. The prover P choses committal l and computes the commit $c=l^2$.
2. The prover computes the responses $r_0=l$ and $r_1=l^e$ where s is a square root of z modulo n .
3. The prover sends the transaction $(c, OT(r_0,r_1))$ to the verifier.
4. Verifier computes the predicate $r^2=c$ if accordingly its secret key the first message is chosen and predicate $r^2=cz$ otherwise. If the computed value is *true*, the verifier accepts otherwise it rejects.

From verifier oblivious transfer public keys, the computationally unbounded dishonest prover \tilde{P} can compute corresponding secret keys and discover the verifier chose. If it corresponds to the first message, the dishonest prover sends the transaction with commit $c=l^2 \bmod n$, $r_0=l$ and arbitrary r_1 , otherwise it sends the transaction with $c=l^2 \tilde{z} \bmod n$, $r_1=l$ and arbitrary r_0 . It follows that the dishonest prover has to be computationally bounded. Such a prover will be successive only if it guesses the verifier chose even it knows the witness.

6 Conclusion Remarks

In this chapter, some novel non-interactive zero-knowledge (NIZK) arguments protocols using non-interactive oblivious transfer (NIOT) have been presented. They are the non-interactive analogues of interactive zero-knowledge proofs with binary and multibit challengers. The key information of used NIOT protocols is adapted from corresponding Bellare – Rivest fractional OT protocols and the

encryption is carried out on ElGamal. They can be realized both in a multiplicative and in an additive group of prime order. It has been shown that through the use of different encryption keys, this implementation can be simplified using a single randomizer for all encryptions in separate session. The proposal allows essentially decrease the soundness probability of NIZK arguments at the same information rate or to increase the information rate at the same soundness probability. The following question remains open: are the security of NIZKOT $_{2^t}^{t+1}$ argument and the Diffie–Hellman problem polynomially equivalent?

Acknowledgement. This research has been supported by Russian Foundation of Basic Research, project 11-01-00792a.

References

- [1] Goldwasser, S., Micali, S., Rackoff, C.: Knowledge Complexity of Interactive Proof Systems. In: Micali, S. (ed.) *Advances in Computing Research: A Research Annual. Randomness and computation*, vol. 5, pp. 73–90 (1986); Extended abstract in 18th STOC, pp. 59–68
- [2] Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: 20th Annual ACM STOC, pp. 103–112 (1988)
- [3] De Santis, A., Micali, S., Persiano, G.: Non-interactive zero-knowledge proof systems. In: Pomerance, C. (ed.) *CRYPTO 1987*. LNCS, vol. 293, pp. 52–72. Springer, Heidelberg (1988)
- [4] Blum, M., De Santis, A., Micali, S., Persiano, G.: Non-interactive zero-knowledge. *SIAM Journal of Computing* 20(6), 1084–1118 (1991)
- [5] De Santis, A., Di Crescenzo, G., Persiano, G.: Randomness efficient non-interactive zero-knowledge (extended abstract). In: Degano, P., Gorrieri, R., Marchetti-Spaccamela, A. (eds.) *ICALP 1997*. LNCS, vol. 1256, Springer, Heidelberg (1997)
- [6] Chase, M., Lysyanskaya, A.: Simulatable VRFs with Application to Multi-Theorem NIZK. In: Menezes, A. (ed.) *CRYPTO 2007*. LNCS, vol. 4622, pp. 303–322. Springer, Heidelberg (2007)
- [7] Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero-knowledge proofs under general assumptions. *SIAM Journal on Computing* 29(1), 1–28 (1999)
- [8] Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero-knowledge proofs are equivalent. In: Vaudenay, S. (ed.) *EUROCRYPT 2006*. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006)
- [9] Fiat, A., Shamir, A.: How to prove yourself: practical solutions of identification and signature problems. In: Odlyzko, A.M. (ed.) *CRYPTO 1986*. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
- [10] Goldwasser, S., Tauman Kalai, Y.: On the (in)security of the Fiat-Shamir paradigm. In: 44th FOCS, pp. 102–115. IEEE Computer Society Press (2003)
- [11] Koblitz, N.: *A Course in number theory and cryptography*. Springer, New York (1994)
- [12] Rabin, M.O.: How to exchange secrets by oblivious transfer. Technical Report TR-81. Aiken Computation Laboratory, Harvard University (1981)

- [13] Blum, M.: How to exchange (secret) keys. *Trans. Computer Systems* 1, 175–193 (1983)
- [14] Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. *Communications of the ACM* 28, 637–647 (1985)
- [15] ElGamal, T.: A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory* IT-31(4):31(4), 469–472 (1985)
- [16] Bellare, M., Rivest, R.L.: Translucent cryptography – an alternative to key escrow, and its implementation via fractional oblivious transfer. MIT/LCS Technical Report 683 (1990)
- [17] Brassard, G., Crépeau, C., Robert, J.M.: Oblivious transfer and intersecting codes. *IEEE Transaction of Information Theory, Special Issue on Coding and Complexity* 42, 1769–1780 (1996)
- [18] Mamontov, A.I., Frolov, A.B.: On one scheme for oblivious transfer of combinations of messages. *PEI Bulletin* 3, 113–119 (2005) (in Russian)
- [19] Mu, Y., Zhang, J., Varadharajan, V.: m out of n oblivious transfer. In: Batten, L.M., Seberry, J. (eds.) *ACISP 2002*. LNCS, vol. 2384, pp. 395–405. Springer, Heidelberg (2002)
- [20] Nyberg, K., Rueppel, R.A.: A new signature scheme based on the DSA giving message recovery. In: *1st ACM Conf. on Computer and Communications Security*, Fairfax, Virginia, pp. 58–61 (1993)
- [21] Frolov, A.: Effective Oblivious Transfer Using Probabilistic Encryption. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability*. AISC, vol. 170, pp. 131–147. Springer, Heidelberg (2012)
- [22] Schnorr, C.-P.: Efficient identification and signature for smart cards. In: Brassard, G. (ed.) *CRYPTO 1989*. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
- [23] Mao, W.: *Modern Cryptography. Theory and practice*. Hewlett Packard Books: Walter Bruce. Hewlett Packard Company (2004)

Virtual Environment for Implementation and Testing Private Wide Area Network Solutions

Mariusz Gola and Adam Czubak

Institute of Mathematics and Informatics,
Opole University
ul. Oleska 48, 45-052 Opole, Poland
{mariusz.gola,adam.czubak}@math.uni.opole.pl

Abstract. In this paper the concept of virtual environment for implementation and testing private Wide Area Network (WAN) solutions is presented. The VMware vSphere virtualization platform is used. The paper presents the ability to reflect the structure of any given WAN topology using Vyatta software routers and VMware virtualization platform and verifies its reliability regarding data transfer. The paper includes a number of performance tests to verify the dependability of the proposed solution and provide a proof-of-concept for the network topology during the Design phase of the PPDIOO methodology, right before the Implementation phase.

1 Introduction

The Wide Area Networks are very important layer 2 and layer 3 technological solutions used for all kinds of business, marketing and general communication applications. In most industries the decision is made to design and implement their own Wide Area Networks to ensure a reliable, secure and fast data exchange between the Enterprise Edge¹ and remote locations. The process of designing from scratch or improving the WANs on one hand requires dedicated optimization algorithms and on the other a reliable testing and simulation environment to ensure, that the design is both theoretically efficient and will serve its purpose in practice.

Topology design of the WAN is one of the most important responsibilities that a network engineer or project manager takes. The PPDIOO methodology [1] is the usually applied method. Proper topology design of WAN saves time, money and company resources. Poorly designed networks result in excessive downtime, unsecured data and inefficient internal and external communication.

Several different formulations of topology design problem can be found in the literature, generally these correspond to [2,4,6,9]:

¹ Enterprise Edge – A framework for designing remote connectivity, a module of the Enterprise Architecture [1].

- Various performance measures according to a given set of criteria;
- Various design variables and imposed constraints.

Typically, the following performance measures are used [6]: average delay and throughput, network reliability and quality of services, cost.

Most of the available research is focused on the modeling stage, the results are presented, sometimes proposed solutions are simulated, but it is rare that the solutions were implemented and tested in practice, i.e. a viable proof-of-concept is not implemented. One reason for that is the lack of adequate laboratory equipped with the required NIC cards for L2 WAN interfaces. The required amount of hardware is also tremendous. To build a medium-size experimental WAN network about a dozen of routers is required. In addition, the installation phase and implementation of solutions for WANs, due to the distance between the individual network locations is time-consuming and impractical in the Design phase. One of possible solutions is to project the proposed solution onto a virtual platform using software routers.

Basic aspects of virtualization like virtual networks, software routers and topology design in virtual environment will be presented in the second section. Performance tests are in the third section. We conclude in the fourth section.

2 Virtualization

Virtualization is a very effective way to reduce IT expenses while boosting efficiency and agility - not just for large enterprises, but for small and midsize businesses as well. Virtualization allows to [5,7-8]:

- Run multiple operating systems and applications on a single computer;
- Consolidate hardware to get vastly higher productivity from fewer servers;
- Save 50% or more on overall IT costs;
- Speed up and simplify IT management, maintenance, and the deployment of new applications.

Virtualization works by enabling multiple operating systems and their applications to run concurrently on a single physical machine or a cloud. These operating systems and applications are isolated from each other within their own secure virtual machines that coexist on a single piece of hardware or a cloud. The virtualization layer maps the physical hardware resources to the virtual machine's resources, so each virtual machine has its own CPU, memory, drives, I/O devices, etc. Virtual machines are a complete equivalent of a standard PC or a server machine. An important feature of virtualization is the ability to easily and intuitively create testing environments. Using virtualization, we can create a test environment in a fairly short time without generating additional costs for hardware and other network equipment. The ease with which we can launch a test environment before considering its deployment in a corporate network makes it a very promising and relatively low cost technology.

With virtualization we can build complex networks, develop, test, and deploy new network ideas - all on a single computer. Other reason, why it is an extremely useful concept is the fact, that the designed WAN is in 'one place', easily available for additional configuration, modifications or version change.

2.1 Virtualization Architecture

A Hypervisor also called Virtual Machine Monitor (VMM) is a piece of software which carries out the process of virtualization. We are dealing with two different kinds of hypervisors. Hypervisor type 1 is called bare-metal hypervisor. In this very case the software is running directly on the hardware, controls all its aspects and monitors the virtual machines. As a result, the client operating system is a virtual machine running on a higher abstract level than the hypervisor. The most commonly used contemporary solutions are VMware vSphere², Citrix XenServer³ and Microsoft Hyper-V⁴. An example of this type of architecture is shown in Figure 1.

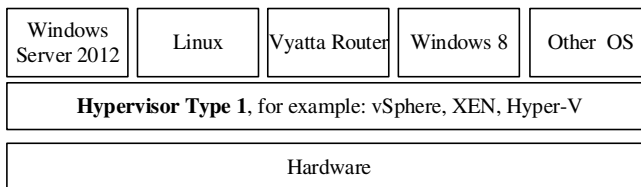


Fig. 1 Hypervisor Type 1 architecture

The second type of hypervisors acts as a conventional application run within the operating system. In this case, installed directly on the hardware and the hosting operating system, the hypervisor is just an application which allows for running virtual machines. So in this case the system running the virtual machine is on the third level of the hardware as shown in Figure 2. Available solutions include VMware Workstation⁵ or Oracle VirtualBox⁶.

In this article we use the VMware vSphere first type hypervisor. First class supervisor is the most efficient since it operates directly on hardware level without an intermediate operating system. VMware vSphere is also the industry-leading virtualization platform for building cloud infrastructures at the moment so this platform was chosen for further investigation.

² For more information please visit: [http:// www.vmware.com](http://www.vmware.com)

³ For more information please visit: [http:// www.citrix.com/xenserver](http://www.citrix.com/xenserver)

⁴ For more information please visit: [http:// http://www.microsoft.com](http://http://www.microsoft.com)

⁵ For more information please visit: [http:// www.vmware.com](http://www.vmware.com)

⁶ For more information please visit: <https://www.virtualbox.org/>

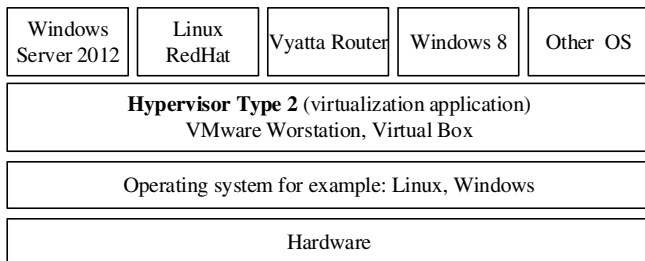


Fig. 2 Hypervisor type 2 architecture

2.2 *Virtual Networks*

On a logical, structural level, virtual networks available in VMware aren't that different from physical networks. vSphere is designed to provide and mimic the functions of a physical network. A virtual network consists of one or more virtual machines connected, so that these are able to send data across the link. A virtual machine can be configured with one or more virtual NICs. Virtual adapters are presented to the guest OS by the virtual machine hardware. These virtual adapters are treated by the guest OS as common network interface cards and will use standard drivers available by default within the guest OS.

Virtual switches allow virtual machines on the vSphere host to communicate with each other using the same protocols used in physical 802.3 Ethernet layer 2 switches. The virtual switch emulates a traditional physical network switch to the extent that it forwards frames within the data link layer and supports basic QoS. A vSphere host may contain multiple virtual switches, each providing more than 1,000 internal virtual ports for virtual machine use. To connect to a vSwitch, a virtual machine must have a virtual NIC (vNIC) mapped to it — just like physical machines can't connect to a network without a network adapter and a switch port. Like real NICs, vNICs have both a MAC address and an IP address. vSwitches can consist of one or more port groups, which describe how the virtual switch should propagate traffic between the virtual network and the virtual machines connected to the specific ports. Network engineers may also use port groups to configure QoS settings like: traffic shaping and bandwidth limitations, NIC failover and many other. Port groups are a very convenient way to achieve WAN links in a virtual environment. The article [3] investigates the state of the art in network virtualization along with the future challenges that must be addressed to realize a viable network virtualization environment useful for commercial and not only academic purposes. Designing and embedding reliable virtual infrastructures is presented in [10].

2.3 *Vyatta Software Router*

Vyatta is a specialized Linux distribution based on Debian, designed to act as a router. It supports all major routing protocols such as RIP, RIP2 OSPF, OSPFv3,

BGP, MP-BGP. It is a mature solution with tools for traffic inspection, filtering and intrusion detection. In addition, it can be configured as a VPN gateway with support for PPTP, L2TP and IPSec and cooperation with RADIUS AAA servers. There are also other functionalities available like: DHCP server, NAT and PAT, QoS, SNMP and many other. The way that Vyatta routers are configured is similar to the syntax implemented in JUNOS on Juniper equipment.

When we install Vyatta onto a standard x86 hardware system, we can create an enterprise grade network appliance that easily scales from DSL to 10Gbps uplinks. Vyatta is also optimized to run in VMware, Citrix XenServer and Hyper-V, providing networking and security services to virtual machines and cloud computing environments. So Vyatta routers are easily implemented in virtual environments and may be duplicated using available clone tools.

2.4 Reflecting the Topology of WAN Networks in a Virtual Environment Using VMware vSphere Platform

As mentioned in the previous section, most of the available research regarding the process of designing WAN networks is focused on the modeling stage, the results are presented, sometimes proposed solutions are simulated, but it seldom that a viable proof-of-concept is implemented, yielding mostly theoretical results in the form of flow tables and a data flow schema. In order to verify the analyzed topology it is best to implement it with a corporate routing configuration and generate excessive data flow in order to measure chosen characteristics of this very topology.

Testing on a virtual platform is economical because it can be carried out without additional hardware costs. It is flexible regarding reconfiguration, since we can easily change bandwidth of any link at any time and practically any other parameter of network at any given moment. Testing results collected from a virtual environment are easier to analyze than results from real-life production network because important information at critical points can be easily logged using industry-standard software to help both the researcher and the network engineer to diagnose the issue. For the experiments a typical network structure topology consisting of 10 nodes and 14 channels connecting the nodes was chosen. For the practical implementation of the topology VMware vSphere virtual environment 5.0 is used. The vSphere is installed on the Fujitsu Primergy RX200 S6 server, equipped with two Quad-Core Intel Xeon E5620. The basic parameters of the virtualization platform are presented in Table 1.

Table 1 Basic parameters of the virtualization platform Fujitsu Primergy RX200 S6

Processor Type	Intel® Xeon® CPU E5620 @ 2.40GHz
Number of sockets	2
Cores per Socket	4
Logical Processor	16

In each of the performed experiments the resources assigned to individual virtual machines were quantified in a way, that their lack would not influence the results. Nevertheless, during the experiments the available resources were monitored as well, since their lack could blur or even corrupt the obtained results.

In Figure 3 the implementation of the network structure is presented. One virtual machine R1 was created and the Vyatta OS installed. Next the required Gigabit Ethernet NICs were assigned to it. This VM was then duplicated to create routers R2 to R9 and the copies accordingly reconfigured. The links WAN01-WAN14 were prepared using virtual switches and port groups available in vSphere. The whole topology was then initiated and tested for connectivity correctness in order to achieve “the golden moment”.

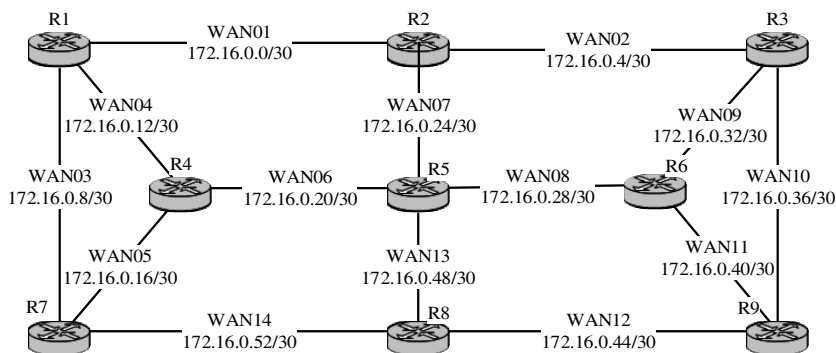


Fig. 3 The implementation of the network topology in VMware vSphere

In order to verify whether the virtual solution is capable of performing real-life functions, the topology was implemented with the following functionalities:

- SAMBA servers were successfully deployed and connected to nodes R1 and R9, in node R1 SAMBA serves as a NT4 domain controller, in location 9 basic folder sharing was activated. Transfer of files sized between 5 and 700MB was successful;
- QoS configuration involving DSCP manipulation in Layer 3 on the border routers R1,R2,R3,R7,R8,R9 and Fair Queue implemented on the output interfaces on routers R4,R5 and R6. Additionally a default WRED mechanism was activated. The congestion was generated by the SAMBA servers.
- OSPF, a dynamic routing protocol was implemented with the default configuration in a single area scenario, default the link costs have not been manipulated.

It must be noted, that the above configuration is only a partial real network setup and it is not clear what kind of issues may arise if other mechanisms, like NAT, PAT, stateful firewall or VRFs were to be implemented.

The topology in Figure 3 allowed for the verification whether the above mentioned network services function in a virtual environment. Since the flows in this configuration are not symmetrical, a different, 1D-topology was created, described in Section 3, for the purpose of experiments.

3 Performance Experiments

The main goal of the experiments was to check the stability and efficiency of virtual network working on a single host computer. While performing the test data was transferred between the server and the client PC through the virtual network, running on one host server. The virtual network consists of Vyatta routers which are connected one by one in a straight line achieving a 1D-topology. In our tests the number of routers varied from 1 to 9. In Figure 4 the test virtual network which consists of five routers is presented. Other virtual networks were built in the same way.

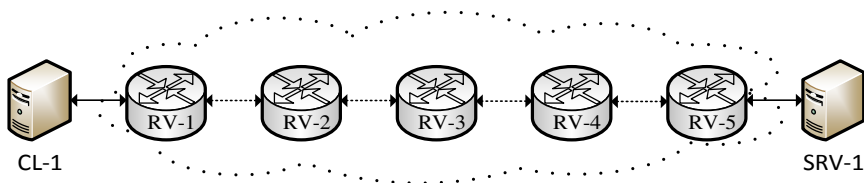


Fig. 4 Schema of a virtual network consisting of five routers

Let $N_{routers}$ be the number of routers in the virtual network;

max_flow is the maximum transfer achieved between the nodes CL-1 and SRV-1
 $single_proc$ is the average processor load in a single virtual router in MHz

$total_proc$ is the total load generated by all the routers running in the virtual network (MHz). The results are shown in Table 2.

Table 2 Experimental results

$N_{routers}$	max_flow	$single_proc$	$total_proc$
1	999	1502	1502
2	916	1407	2814
3	803	1324	3972
4	724	1222	4888
5	645	1134	5670
6	525	1004	6024
7	443	895	6265
8	374	821	6568
9	312	743	6687

In the Figure 5 dependence between max_flow and $N_routers$ is presented. This dependence is clearly linear in nature.

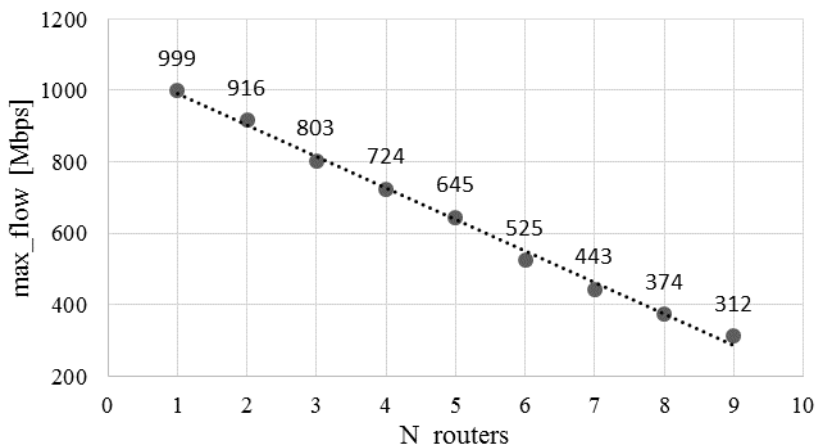


Fig. 5 Dependence of max_flow on $N_routers$

The parameters of this linear function

$$max_flow = a * N_routers + b$$

were designated, yielding accordingly: $a = -88.2$ and $b = 1079$.

Another experiment was performed, which was to determine the number of routers when the transfer drops below the accepted 100Mbit/s throughput. This amount equals to 9 routers.

In Figure 6 the relationship between $single_proc$ and max_flow is presented.

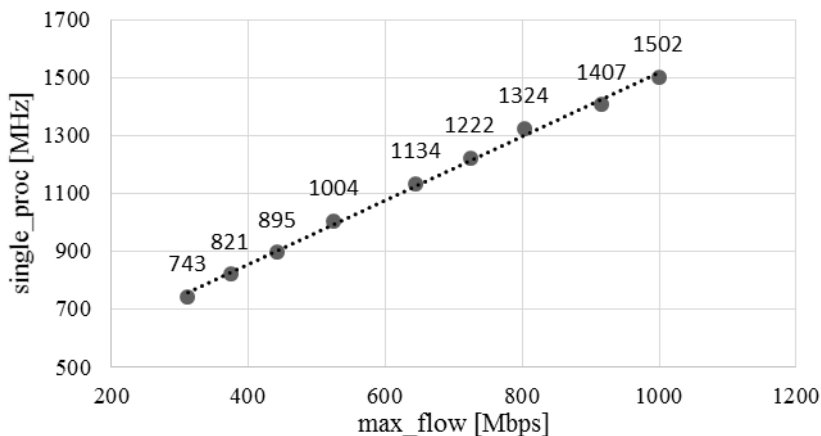


Fig. 6 Dependence of $single_proc$ on max_flow

This dependence we approximate by linear function. The parameters of this function

$$single_proc = a * max_flow + b$$

were estimated as follows: $a = 1.1$, $b = 411$.

The results allow for the assessment of the resources used by the virtual routers regarding the amount of traversing traffic.

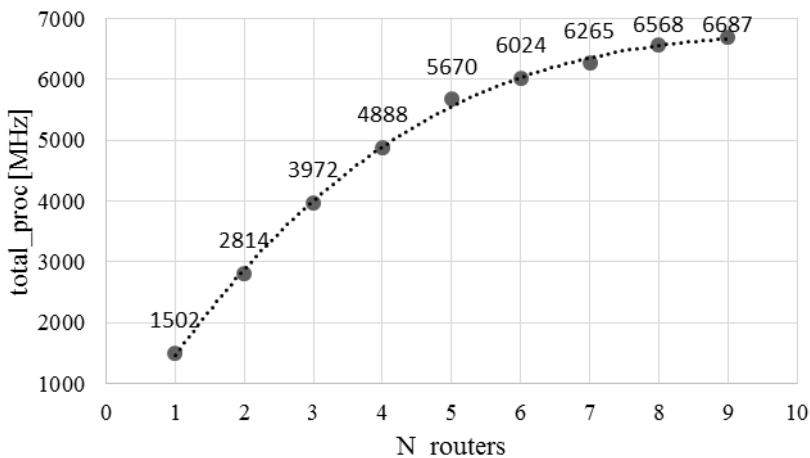


Fig. 7 Dependence of *total_proc* on *N_routers*

In Figure 7 the dependence of *total_proc* on *N_routers* is presented. This dependence we approximate by a third degree polynomial:

$$total_proc = a * N_routers^3 + b * N_routers^2 + c * N_routers + d$$

With the coefficients: $a = 5.2854$, $b = -172.2$, $c = 1891.7$, $d = -263.33$.

During our experiments we noticed that we can easily and comfortably use up to nine virtual routers located on single host computers with channel bandwidth up to 200 Mbps. Transfer rate which is close to 100 Mbps is sufficient for most private Wide Area Networks. Further addition of virtual routers resulted in decrease in network performance according to the trend depicted in Figure 5.

4 Conclusions

The results presented in this paper confirm the assumption, that a virtual environment is suitable for performing proof-of-concept design tasks of private WAN networks. The performed experiments show, that with the increasing number of WAN routers the overall throughput drops linearly. The server used for performed experiments allowed for running a limited number of virtual routers but the available bandwidth drops accordingly.

With the link capacities determined it is easy to predict up to how many virtual routers a virtual environment will be able to run.

The experiments also confirmed the sheer ease of use of the virtual environment for testing purposes.

References

- [1] Bruno, A.: CCDA 640-864 Official Cert Guide, 4th edn., pp. 640–864. Cisco Press, Indianapolis (2011)
- [2] Piech, H., Czubak, A.: The Network Balance Realized by Routing Organization System. In: O’Shea, J., Nguyen, N.T., Crockett, K., Howlett, R.J., Jain, L.C. (eds.) KES-AMSTA 2011. LNCS (LNAI), vol. 6682, pp. 425–435. Springer, Heidelberg (2011)
- [3] Chowdhury, M., Boutaba, R.: Network Virtualization: State of the Art and Research Challenges. *IEEE Communications Magazine* 47, 20–26 (2009), doi:10.1109/MCOM.2009.5183468
- [4] Gola, M., Kasprzak, A.: Exact and Approximate Algorithms for Two-Criteria Topological Design Problem of WAN with Budget and Delay Constraints. In: Laganá, A., Gavrilova, M.L., Kumar, V., Mun, Y., Tan, C.J.K., Gervasi, O. (eds.) ICCSA 2004. LNCS, vol. 3045, pp. 611–620. Springer, Heidelberg (2004)
- [5] Haletky, E.: VMware ESX and ESXi in the Enterprise: Planning Deployment of Virtualization Servers. Prentice Hall (2011)
- [6] Kasprzak, A.: Topological Design of the Wide Area Networks. Wroclaw University of Technology Press, Wroclaw (2001)
- [7] Lowe, S.: Mastering VMware vSphere 5. Sybex (2011)
- [8] Portnoy, M.: Virtualization Essentials. Sybex (2012)
- [9] Walkowiak, K.: A Flow Deviation Algorithm for Joint Optimization of Unicast and Anycast Flows in Connection-Oriented Networks. In: Gervasi, O., Murgante, B., Laganà, A., Taniar, D., Mun, Y., Gavrilova, M.L. (eds.) ICCSA 2008, Part II. LNCS, vol. 5073, pp. 797–807. Springer, Heidelberg (2008)
- [10] Yeow, W.L., Westphal, C., Kozat, U.C.: Designing and embedding reliable virtual infrastructures. *ACM SIGCOMM Computer Communication Review* 41(2), 57–64 (2011), doi:10.1145/1851399.1851406

Optimization of Privacy Preserving Mechanisms in Mining Continuous Patterns

Marcin Gorawski^{1,2} and Pawel Jureczek²

¹ Wrocław University of Technology, Institute of Computer Science,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
Marcin.Gorawski@pwr.wroc.pl

² Silesian University of Technology, Institute of Computer Science,
Akademicka 16, 44-100 Gliwice, Poland
{Marcin.Gorawski, Pawel.Jureczek}@polsl.pl

Abstract. This work presents a new Apriori-like algorithm called CPMPP (Continuous Pattern Mining with Privacy Preservation) for exploring long continuous sequences in a distributed environment with privacy preservation. Given the fact that a cryptography-based technique may consume a considerable amount of time when enciphering data using a commutative cipher, a short form of locally frequent sequences has been applied to improve the system performance. In this approach, the length of locally frequent sequences does not exceed two items. In consequence, the proposed algorithm reduces the number of expensive cryptographic operations required to obtain the result. The conducted experiments show a significant performance increase with the increase of the length of generated patterns.

1 Introduction

Data mining on data extracted from several independent systems gives a more complete view of important relationships in the data. The knowledge gained in this way may bring benefits to all participants involved in the analysis. However, due to privacy reasons, an exploration cannot always be performed on one data store located at a trusted site. When the privacy of parties involved in the analysis is a priority, we can use semi-honest model. In this model, a trusted site is replaced with the secure multi-party protocol [1-4, 15]. The goal of this protocol is to hide the identity of the owner of a particular item. Moreover, due to the existence of many large sequence databases with huge amounts of data (e.g., Moving Objects Databases and Trajectory Data Warehouses [12]) and the growing need for highly scalable distributed systems, it is crucial to investigate efficient methods for distributed mining of sequences with privacy preservation.

The solution presented in this work is based on the CDKSU (Secure Union with Common Decrypting Key) algorithm described in [5]. This algorithm is extended in order to handle problems which have not been investigated in the original

concept. Our new approach provides a variant of the CDKSU algorithm designed to explore continuous patterns [6, 13-14] in an efficient way. A significant performance gain is achieved by using a new distributed Apriori-like algorithm called CPMPP (Continuous Pattern Mining with Privacy Preservation) and a dedicated data structure. A key feature of the algorithm is that it uses a compact form of continuous sequences. Furthermore, a dedicated data structure aggregates sequences in an index tree. This index improves the performance of the support computation.

The rest of the chapter is structured as follows. Section 2 presents an overview of the CDKSU algorithm. Section 3 gives a general overview of the new approach. In Section 4, we discuss the impact of different parameters on the runtime of the CPMPP algorithm. Section 5 summarizes our work.

2 CDKSU Algorithm

In this work we assume that the different sites (data owners) do not want to expose all their data (which might be sensitive or valuable), but they still want to mine the union of their databases. Generally speaking, data owners combine their efforts to mine sequential patterns but by analyzing the union of data – stored at one site – one could deduce something about data sources. Hence, all patterns are encrypted by all parties and since only the full encryption allows for decryption using the common key, it is not possible to deduce which data source gave rise to which patterns.

2.1 Overview of CDKSU Algorithm

The CDKSU algorithm is a distributed association rules mining algorithm for horizontally partitioned data, which preserves data privacy in semi-honest model. In this model it is assumed that each site follows honestly the protocol but may try to deduce information about the other site's private data by analyzing information received as a result of performing the protocol. In this model a commutative encryption is commonly used.

Notice that our commutative ciphers work on fixed-length units called blocks. It means that if we want to encrypt a long plaintext then it has to be split up into blocks and then these blocks should be encrypted separately. Obviously, the longer a message is, the more time is needed to encrypt the whole message. The similar situation occurs when a ciphertext is decrypted.

In order to reduce the number of expensive decryption operations, the CDK (Common Decryption Key) can be calculated for a commutative cipher to decrypt every ciphertext only once. This approach has been applied in the CDKSU algorithm.

Let us assume that a transaction database DB is partitioned horizontally over n sites S_1, S_2, \dots, S_n in such a way that DB_i resides at site S_i . Every transaction

in DB_i is a set of items. An itemset that contains k items is called k -itemset. The support of a k -itemset is defined as the ratio of the number of transactions containing it to the total number of transactions in DB . An itemset is considered to be frequent if its support is no less than a user defined support threshold. Moreover, the locally frequent itemset at site S_i is an itemset that is frequent with respect to DB_i . Notice that a locally frequent itemset may not be globally frequent, i.e., it may not be frequent with respect to DB .

Furthermore, each site S_i has its own secret key which is not shared with any other site and one site is selected as site D which is responsible for decrypting data.

The main steps of the CDKSU algorithm are:

1. Each party computes the intersection of the set of locally frequent itemsets and the set of globally frequent itemsets, and then generates candidate itemsets according to the Apriori property [7]. Next, the support for each candidate is counted and finally locally frequent itemsets are found.
2. After all locally frequent itemsets are discovered by a party, they are encrypted using the secret key and sent to a next party.
When a party receives itemsets from other party, these itemsets are encrypted again and sent to a next party. This process is repeated until all itemset are encrypted by all parties.
3. A site, which is not site D , determines the union of all locally frequent itemsets.
4. CDK is calculated in order to decrypt itemsets found in step 3.
5. Using the itemsets from the previous step, globally frequent itemsets are found and sent to all parties. The algorithm terminates when there are no globally frequent itemsets.

As we can see, CDK decreases the number of expensive decryption operations required to obtain the result, since every itemset is decrypted only once.

Notice that the description of CDKSU has been simplified to improve readability. For more details please refer to [5].

2.2 Contribution and Security Issues

In our work, we focus on data privacy preservation in sequence databases. In this view, the main difference between the original proposition (i.e., CDKSU) and CPMPP is that the latter works on continuous sequences. Moreover, CPMPP uses a compact form of continuous sequences. To the best of our knowledge, there has not been any work addressing these issues. It is important to notice that frequent pattern mining is essentially different from continuous pattern mining, and what is more, these mining processes produce patterns that have different properties (e.g., order and continuous).

CPMPP described in the rest of the work does not violate the CDKSU cryptography-based scheme. Our approach can be seen as a variant of the CDKSU scheme with a new version of the Apriori algorithm. Furthermore,

privacy-preserving properties of our new proposal do not need to be evaluated, since no new information is disclosed to any party.

3 CPMPP Algorithm

As mentioned in the previous section, our cryptosystem is based on the CDKSU algorithm which uses a commutative encryption. In this cryptosystem, each site involved in the collaboration encrypts data blocks using a private key. Moreover, only one site decrypts fully encrypted data to obtain the final result. In order to speed up CDKSU, a novel approach has been introduced.

The main idea is based on the observation that the more sites are involved in secure pattern mining (assuming the same support threshold), the more time is needed to complete the task (recall Subsection 2.1). A similar situation occurs when the average length of frequent patterns increases. In this view, we present an approach that limits the length of locally frequent sequences broadcasted to participating sites. By reducing the length of sequences, the number of data blocks that are enciphered and deciphered decreases. According to the CPMPP algorithm, the length of a candidate will never exceed 2. That also means that every locally frequent sequence will be contained in a single data block.

3.1 Definitions

In this subsection we define the terminology used in the rest of the chapter:

Definition 1: Continuous sequence

Given a distinct set of items $E=\{e_1, e_2, \dots, e_n\}$, a continuous sequence is a sequence of items $\langle a_1 a_2 \dots a_m \rangle$ where $a_i \in E$ ($1 \leq i \leq m$) and for any two items a_i and a_j ($i \neq j$) we have $a_i \neq a_j$.

A continuous k -sequence is a sequence with length k , where the length of a continuous sequence is the number of items in it.

Where it is clear from the context we will omit the word *continuous*.

Definition 2: Containing \subseteq_{cs} one continuous sequence in other continuous sequence:

A continuous sequence $s_1=\langle b_1 b_2 \dots b_m \rangle$ is a continuous subsequence of a continuous sequence $s_2=\langle a_1 a_2 \dots a_n \rangle$ ($n \geq m$), denoted as $s_1 \subseteq_{cs} s_2$, if for a certain integer i $b_1=a_i$, $b_2=a_{i+1}$, ..., $b_m=a_{i+m-1}$. On the other hand, the sequence s_2 is a supersequence of s_1 .

Definition 3: Support of a continuous sequence

The relative support $supp_{cs}$ of a continuous sequence s is calculated as:

$$supp_{cs}(s) = \frac{\text{number of sequences containing } s}{\text{total number of sequences}} * 100\%$$

Similarly, the number of sequences that contain s is called the absolute support $asupp_{cs}$ of s .

For instance, let a sequence database contain two continuous sequences $\langle A F G Z \rangle$ and $\langle A B F G \rangle$. The relative support $supp_{cs}$ of the continuous sequence $\langle A F G \rangle$ is 50%, since only $\langle A F G Z \rangle$ matches $\langle A F G \rangle$.

Definition 4: Frequency of a continuous sequence

A continuous sequence s is called frequent if its support $supp_{cs}$ ($asupp_{cs}$) is no less than a threshold $minSup$ given by a user, i.e.:

$$supp_{cs}(s) \geq minSup \text{ (or } asupp_{cs}(s) \geq minSup)$$

Definition 5: Locally frequent continuous sequence:

A continuous sequence s is called locally frequent at site S_i if its support is no less than a threshold $minSup$ with respect to DB_i .

Definition 6: Globally frequent continuous sequence:

A continuous sequence s is called globally frequent if its support is no less than a threshold $minSup$ with respect to DB , where $DB = \bigcup_{i=1}^n DB_i$ and n is the number of parties.

Definition 7: Extension

The extension ext is a 2-sequence defined as $ext = \langle a_1 a_2 \rangle$ where a_1 and a_2 belong to the set of items $E = \{e_1, e_2, \dots, e_n\}$ and $a_1 \neq a_2$.

For instance, the extension $\langle B Y \rangle$ means that it starts with the item B and B is followed by Y .

Definition 8: Placeholder

The placeholder x for a continuous sequence is a unique identifier which is defined as $x = \langle a_1 a_2 \rangle$ (i.e., $\langle a_1 a_2 \rangle$ can be replaced by x and vice versa).

For instance, using the placeholder $x = \langle a_1 a_2 \rangle$ the continuous sequence $s = \langle a_1 a_2 a_3 \rangle$ can be written as $s = \langle x a_3 \rangle$.

3.2 CPMPP

CPMPP assumes that one auxiliary item (i.e., placeholder) is added to every globally frequent sequence (step 5 in Subsection 2.1) when the length of a globally frequent sequence equals 2. The placeholder is a unique identifier of the globally frequent sequence and, which is extremely important, it does not reveal any new

information to the attacker. Additionally, during generation of candidate sequences (step 1 in Subsection 2.1), each site replaces the first two items of candidates with corresponding placeholders.

Furthermore, CPMPP is a version of the Apriori algorithm [7] and works on three main sets. The first set *GFS* contains globally frequent sequences found in the previous iteration of the algorithm, the second set *LFS* holds locally frequent sequences, while the last set *LFE* contains locally frequent extensions. The set of locally frequent sequences *LFS* is sent to a next party (step 2 in Subsection 2.1). However, the set of extensions *LFE* is stored locally and is never sent to any party (i.e., is not shared with the parties and remains private).

As other distributed versions of the Apriori algorithm (e.g., FDM [8]), the CPMPP iteration consists of a sequence of steps, including generation of candidates, support computation, local pruning and result forwarding. A main difference is that the generation of candidates in CPMPP is based on globally frequent sequences, locally frequent sequences and extensions. More precisely, at the beginning of each iteration, the intersection of the first two sets is found. If the intersection set *GLS* is not empty, candidates are generated by taking into account both *GLS* and *LFE*.

The extensions are in some sense a history of the previous iteration and themselves cannot generate candidates. Nevertheless, in order to generate all possible candidates they have to be analyzed.

Figures 1-3 show successive iterations of the CPMPP algorithm at site 0. Please assume that site 0 has three sequences $s_1=\langle A B C \rangle$, $s_2=\langle B C D Y J \rangle$ and $s_3=\langle I C D Y J \rangle$ in DB_0 , and the (absolute) minimum support $minSup$ equals 2. In order to simplify our description, we also assume, without any loss of generality, that locally frequent sequences are also frequent globally. Because the determination of globally frequent 1-sequences is similar to that in [5], it will be omitted here. The subscript of a set denotes an iteration number.

GFS	Candidates					LFS₁	LFE₁
$\langle B \rangle$	$\langle B C \rangle^+$	$\langle C B \rangle^-$	$\langle D B \rangle^-$	$\langle J B \rangle^-$	$\langle Y B \rangle^-$	$\langle B C \rangle^+$	$\langle C J \rangle^*$
$\langle C \rangle$	$\langle B D \rangle^-$	$\langle C D \rangle^+$	$\langle D C \rangle^-$	$\langle J C \rangle^-$	$\langle Y C \rangle^-$	$\langle C D \rangle^+$	$\langle C Y \rangle^*$
$\langle D \rangle$	$\langle B J \rangle^-$	$\langle C J \rangle^*$	$\langle D J \rangle^*$	$\langle J D \rangle^-$	$\langle Y D \rangle^-$	$\langle D Y \rangle^+$	$\langle D J \rangle^*$
$\langle J \rangle$	$\langle B Y \rangle^-$	$\langle C Y \rangle^*$	$\langle D Y \rangle^+$	$\langle J Y \rangle^-$	$\langle Y J \rangle^+$	$\langle Y J \rangle^+$	
$\langle Y \rangle$							

Fig. 1 First iteration of CPMPP

Based on globally frequent 1-sequences (Fig. 1), 2-candidates are generated. This is done as in the AprioriAll algorithm (details can be found in [9]). The next step is to count the support by scanning the index (index stores all sequences and is used to compute the support of a given sequence in an efficient way).

A local candidate may be in one of 3 states (+, *, -). A plus (+) means that a sequence is a frequent continuous sequence. An asterisk (*) indicates that a

sequence is an extension. A minus (-) is used to denote an infrequent sequence. For example, the candidate $\langle B C \rangle$ is considered to be a frequent continuous sequence since C occurs directly after B and there are two sequences s_1 and s_2 containing this pattern. The candidate $\langle C Y \rangle$ is an extension because Y is located after C in two sequences. The candidate $\langle C B \rangle$ is considered to be infrequent, since the candidate order is not preserved in the sequences in DB_0 .

After examining all the candidates, locally frequent sequences are sent to other parties, while the set of extensions is locally stored.

It should be noted that a globally frequent sequence can also be an extension.

GFS ₁	LFE ₁	Candidates		LFS ₂	LFE ₂
$\langle B C X1 \rangle$	$\langle C J \rangle^*$	$\langle X2 J \rangle^*$	$\langle X3 J \rangle^+$	$\langle X2 Y \rangle^+$	$\langle X2 J \rangle^*$
$\langle C D X2 \rangle$	$\langle C Y \rangle^*$	$\langle X2 Y \rangle^+$		$\langle X3 J \rangle^+$	
$\langle D Y X3 \rangle$	$\langle D J \rangle^*$				
$\langle Y J X4 \rangle$					

Fig. 2 Second iteration of CPMPP

Fig. 2 depicts the next iteration. All globally frequent 2-sequences found according to the minimum support threshold are presented on the left side of the figure. According to previous assumptions, every globally frequent sequence has an additional item (placeholder) that uniquely identifies it. In Fig. 2, placeholders are $X1$, $X2$, $X3$ and $X4$. A placeholder will replace the first two items of a new candidate sequence (more detailed descriptions can be found in the next subsection).

The generation of candidates in Fig. 2 is different from that presented in the previous iteration. First, every candidate should start with a 2-prefix that is present in the set of globally frequent sequences. Second, a candidate can be generated by joining a globally frequent sequence with another one or an extension. Third, two sequences can be joined if they have the same first item. Furthermore, 2-prefix of a candidate is replaced with a corresponding placeholder.

For example, consider the globally frequent sequence $\langle C D X2 \rangle$. The item $X2$ is a placeholder to denote the first two items C and D . We may not join $\langle C D X2 \rangle$ with any other globally frequent sequence as they do not start with C . However, we can join $\langle C D X2 \rangle$ with the extensions: $\langle C J \rangle$ and $\langle C Y \rangle$. As a result, two sequences are generated $\langle C D J \rangle$ and $\langle C D Y \rangle$. Finally, we have to replace all occurrences of 2-prefix $\langle C D \rangle$ with $X2$. The other globally frequent sequences are processed in a similar way. The remaining steps are the same as in the first iteration.

The last iteration is pictured in Fig. 3.

GFS_2	LFE_2	Candidates	LFS_3		
$\langle X2 Y X5 \rangle$	$\langle X2 J \rangle^*$			$\langle X5 J \rangle^+$	$\langle X5 J \rangle^+$
$\langle X3 J X6 \rangle$					

Fig. 3 Third iteration of CPMPP

To clarify the uncertainty, the derivation of $\langle X5 J \rangle$ is as follows. Assuming that $X2 = \langle C D \rangle$ and $X5 = \langle X2 Y \rangle$ (see Figures 2 and 3), we have $\langle X2 Y J \rangle$ and finally $\langle C D Y J \rangle$. Information about the transformations of extensions are stored by every party.

3.3 Pseudocode

In this subsection, a generic pseudocode of the CPMPP algorithm is presented. The descriptions of variables are listed below:

- $GFS_{i,k}$ – set of globally frequent continuous sequences supported by all sites at the k th iteration. The member m of $GFS_{i,k}$ can be seen as a triple $\langle s_1 s_2 x_3 \rangle$, where $\langle s_1 s_2 \rangle$ is a globally frequent sequence and x_3 is its placeholder. $GFS_{i,k}$ is fully decrypted.
- $GLS_{i,k}$ – set of globally frequent continuous sequences locally supported at site S_i .

Algorithm 1. CPMPP

```

1)  $RS = RS \cup deriveResults(GFS_{i,k-1}, TS_{i,k-1})$ 
2)  $TS_{i,k} = retrieveTransformations(GFS_{i,k-1}, TS_{i,k-1})$ 
3)  $GLS_{i,k} = GFS_{i,k-1} \cap LFS_{i,k-1}$ 
4) if  $GLS_{i,k}$  contains 1-sequences then
    $CS_{i,k} = retrieve2SeqFromIndex(GLS_{i,k})$ 
   goto step 7
5) for each  $\langle s_1 s_2 x_3 \rangle \in GLS_{i,k}$  do
   for each  $\langle s_4 s_5 x_6 \rangle \in GLS_{i,k}$  do
     if  $s_1 = s_4$  and  $s_2 \neq s_5$  then
        $CS_{i,k} = CS_{i,k} \cup \langle x_3 s_5 \rangle$ 
6) for each  $\langle s_1 s_2 x_3 \rangle \in GLS_{i,k}$  do
   for each  $\langle s_4 s_5 \rangle \in LFE_{i,k-1}$  do
     if  $s_1 = s_4$  and  $s_2 \neq s_5$  then
        $CS_{i,k} = CS_{i,k} \cup \langle x_3 s_5 \rangle$ 
7) for each  $c \in CS_{i,k}$  do
   if  $c$  is frequent continuous sequence then
      $LFS_{i,k} = LFS_{i,k} \cup c$ 
   else if  $c$  is frequent extension then
      $LFE_{i,k} = LFE_{i,k} \cup c$ 

```

Fig. 4 CPMPP algorithm

- $LFS_{i,k}$ – set of locally frequent continuous sequences supported at site S_i at the k th iteration.
- $CS_{i,k}$ – set of candidates at site S_i at the k th iteration.
- $LFE_{i,k}$ – set of locally frequent extensions supported at site S_i at the k th iteration.
- $TS_{i,k}$ – set of extension transformations at the k th iteration.
- RS – result set.

The description of Algorithm 1 is as follows. In step 1, the set of globally frequent sequences is added to the result set RS . Since in later iterations patterns contain placeholders, it is needed to derive the final forms of those patterns by replacing placeholders with appropriate items. For this purpose, the *deriveResults* function is implemented. In step 2, the *retrieveTransformations* function prepares the set of placeholders for use in the next iteration. In step 3, globally frequent sequences are narrowed to only those locally supported. Steps 5-6 show how candidate sequences are generated, and step 7 how frequent sequences and extensions are obtained. In the case when the length of a globally frequent sequence is 1, candidates are generated directly from the index (see step 4). This narrows down the number of possible candidates and thus greatly speeds up the local processing time. The index is also used to count the frequency of candidates. If $LFS_{i,k}$ is not empty, the party encrypts its locally frequent sequences and sends them to other party.

4 Experiments

In the experiments, we compared two algorithms CPMPP and GSP [10]. The GSP algorithm is equivalent to the CPMPP algorithm, in the sense that it allows one to generate the same result set. Both algorithms are embedded into CDKSU, but the new approach is only applied in CPMPP. A commutative encryption scheme used in the experiments was based on the Elliptic Curve Pohlig-Hellman cipher with prime239v1 parameters [11].

The experiments were carried out for three nodes located on three separate computers connected by a local area network. Each of those nodes had a sequence database consisting of 10k sequences with 629 distinct items. The average length of sequences was 14.2 and the average deviation was 0.5. All the computers were equipped with Intel Core 2 Quad Q9450, 2GB of RAM and Samsung HD252HJ hard drive.

Since the generation of 2-candidates is a time-consuming process, both algorithms fetch locally frequent 2-sequences directly from the index. The support for every sequence is also determined using the index. It should be emphasized that the index of a party is never sent to the other parties.

The measurements of runtime began after a node received globally frequent 1-sequences. Our testing suite concerned measurements of the runtime under different minimum support values.

The runtime of the algorithms for different values of the minimum support and patterns is shown in Fig. 5. The suffix *en* means that encryption is on, and *no* that no encryption is present. The results show that CPMPP is slightly better when the encoding is turned off. However, when the encoding is enabled, performance increases significantly.

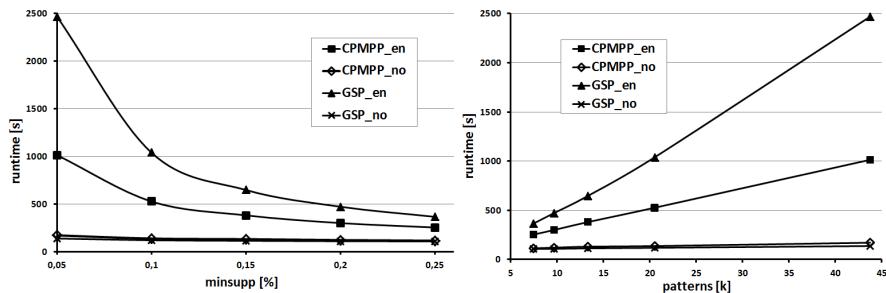


Fig. 5 Runtime vs. minimum support (on the left) and runtime vs. patterns (on the right)

Fig. 6 shows the number of data blocks that are encrypted using the commutative ciphers. The number of blocks increases with decreasing the minimum support, as the algorithms generate more patterns. Moreover, the length of patterns also increases.

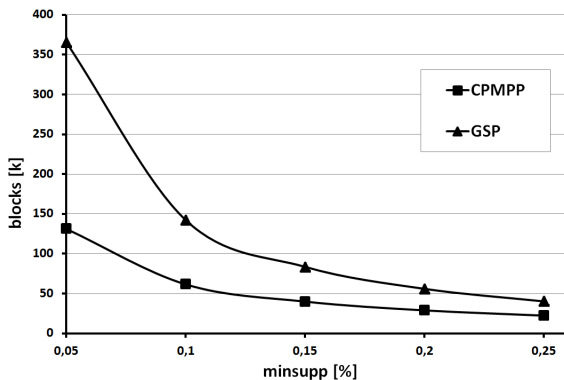


Fig. 6 The number of encrypted blocks vs. minimum support

The performance gain is shown in Fig. 7. This figure shows that CPMPP performs extremely well in comparison with GSP.

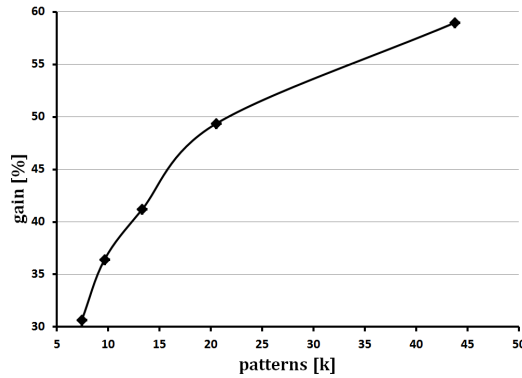


Fig. 7 Performance gain vs. patterns

5 Conclusion and Future Work

Many companies and organizations collaborate to get the benefits of mining their data using privacy-preserving techniques [16, 17]. The cryptography-based schemes found in those techniques are often characterized by low efficiency. In order to improve the efficiency of systems that are based on those schemes we present the CPMPP algorithm for mining continuous sequences in a distributed environment with privacy preservation.

The CPMPP algorithm allows reducing the number of data blocks that are encrypted and decrypted using a commutative cipher. The experiments show that reduction of operations improves performance significantly.

The future research will include further optimization of the CPMPP algorithm. We expect it can be improved by using a novel index-based approach that will find locally frequent sequences in only one step.

References

- [1] Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game. In: Proc. of the 19th Annual ACM Symposium on Theory of Computing, STOC 1987, pp. 218–229. ACM, New York (1987)
- [2] Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639–644 (2002)
- [3] Clifton, C., Kantarcioglu, M., Vaidya, J., Lin, X., Zhu, M.Y.: Tools for privacy preserving distributed data mining. SIGKDD Explor. Newsl. 4(2), 28–34 (2002)
- [4] Kantarcioglu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. Trans. Knowl. Data Eng. 16(9), 1026–1037 (2004)

- [5] Gorawski, M., Siedlecki, Z.: Optimization of Privacy Preserving Mechanisms in Homogeneous Collaborative Association Rules Mining. In: 6th International Conference on Availability, Reliability and Security, pp. 347–352 (2011)
- [6] Gorawski, M., Jureczek, P.: Extensions for Continuous Pattern Mining. In: Yin, H., Wang, W., Rayward-Smith, V. (eds.) IDEAL 2011. LNCS, vol. 6936, pp. 194–203. Springer, Heidelberg (2011)
- [7] Agrawal, R., Srikant, R.: Fast algorithm for mining association rules. In: The International Conference on Very Large Data Bases, pp. 487–499 (1994)
- [8] Cheung, D.W., Han, J., Ng, V.T., Fu, A.W., Fu, Y.: A fast distributed algorithm for mining association rules. In: PDIS: Int. Conf. on Parallel and Distributed Information Systems (1996)
- [9] Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the 8th IEEE International Conference on Data Engineering, pp. 3–14 (1995)
- [10] Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 1–17. Springer, Heidelberg (1996)
- [11] Qu, M.: Standards for efficient cryptography sec 2: Recommended elliptic curve domain parameters (2010)
- [12] Kozielski, S., Wrembel, R. (eds.): New Trends in Data Warehousing and Data Analysis. Annals of Information Systems, vol. 3. Springer (2009)
- [13] Gorawski, M., Jureczek, P., Gorawski, M.: Exploration of continuous sequential patterns using the CPGrowth algorithm. In: Nguyen, N.T., Zgrzywa, A., Czyżewski, A. (eds.) Advances in Multimedia and Network Information System Technologies. AISC, vol. 80, pp. 165–172. Springer, Heidelberg (2010)
- [14] Gorawski, M., Jureczek, P.: Continuous Pattern Mining Using the FCPGrowth Algorithm in Trajectory Data Warehouses. In: Graña Romay, M., Corchado, E., Garcia Sebastian, M.T. (eds.) HAIS 2010, Part I. LNCS (LNAI), vol. 6076, pp. 187–195. Springer, Heidelberg (2010)
- [15] Gorawski, M., Stachurski, K.: On efficiency and data privacy level of association rules mining algorithms within parallel spatial data warehouse. In: Proceedings of the the First International Conference on Availability, Reliability and Security, pp. 936–943 (2006)
- [16] Gorawski, M., Bularz, J.: Protecting private information by data separation in distributed spatial data warehouse. In: ARES 2007, pp. 837–844. IEEE CS (2007)
- [17] Gorawski, M., Morzy, T., Wrembel, R.: Special Issue on: Techniques of Advanced Data Processing and Analysis Introduction. Control and Cybernetics 38(1) (2009)

Technical and Program Aspects on Monitoring of Highway Flows (Case Study of Moscow City)

M.G. Gorodnichev¹ and A.N. Nigmatulin²

¹ Moscow Technical University of Communications and Informatics,
111024 Moscow Aviamotornaya 8-a
gorodnichev@hotmail.com

² Moscow State Automobile and Road Technical University,
125319, Moscow, Leningradskiy Avenue 64
a.n.nigmatulin@yandex.ru

Abstract. The article describes validation and testing of traffic monitoring technology for models creation and forecast of traffic characteristics on traffic city network (based on case study of Moscow). We discuss one of collect data method based on network of microwave radar SSHD, installed in Moscow last year. Acquiring data allow to analyze of traffic jam causes and to evaluate approximately economic losses for typical junctions of road network.

1 Introduction

Nowadays the state of transport network in Moscow (Russia) is caused by the following factors: (1) not well established street & road network, including low quantity of road junctions; (2) very rapidly increasing number of vehicles on the roads; (3) distinctive features of driving – drivers' mentality of citizens and guests. Significant scientific and engineering resources are deployed to research all the possible application of leading existing traffic models [5],[7],[10],[11] in order to predict and manage the traffic flows in Moscow.

Without any exceptions all the approaches in modeling of complex socio-technical systems imply intellectual monitoring, in other words the system of measurement and primary automatic processing of main traffic features based on the example of Moscow road network.

In 2010-2012 DTC¹ of Moscow Government purchased, set up and tested numerous (>2000) microwave radars SmartSensor HD (Legacy) (SS-125). Thus, there is a need to define the optimal quantity of radars and recommend the placements for installation of the device. Moscow State Automobile and Road Technical University (MADI), situated on the roadside of one of the main highways in Moscow road network (Fig. 1), had an opportunity to study the operating quality of the typical device and automatically got the access to all the visual information about traffic on 10 lanes with the processing rights. Profound experimental and

¹ Department of Transport and Communication.

analytical studies were conducted to define the possible application of such monitoring system – on the example of typical device (radar) of the system – to the modeling and forecasting of traffic in Moscow.

The objective of the article is to reveal the pros and cons of data collection about traffic flows with the help of microwave radars SSHD. It is necessary to define the limitations of this method and propose alternative methods of automated monitoring for the operating modes, in which SSHD shows the highest deviations. Based on that it will be possible to define the dynamics of traffic features and analyze the causes of intensive traffic, taking into account the distinctive features of drivers' behavior in Moscow. It is important to show practical value – define economic losses of traffic in order to actualize the problem even better. The outcomes of the study and the conclusions on the traffic state are described in this article.

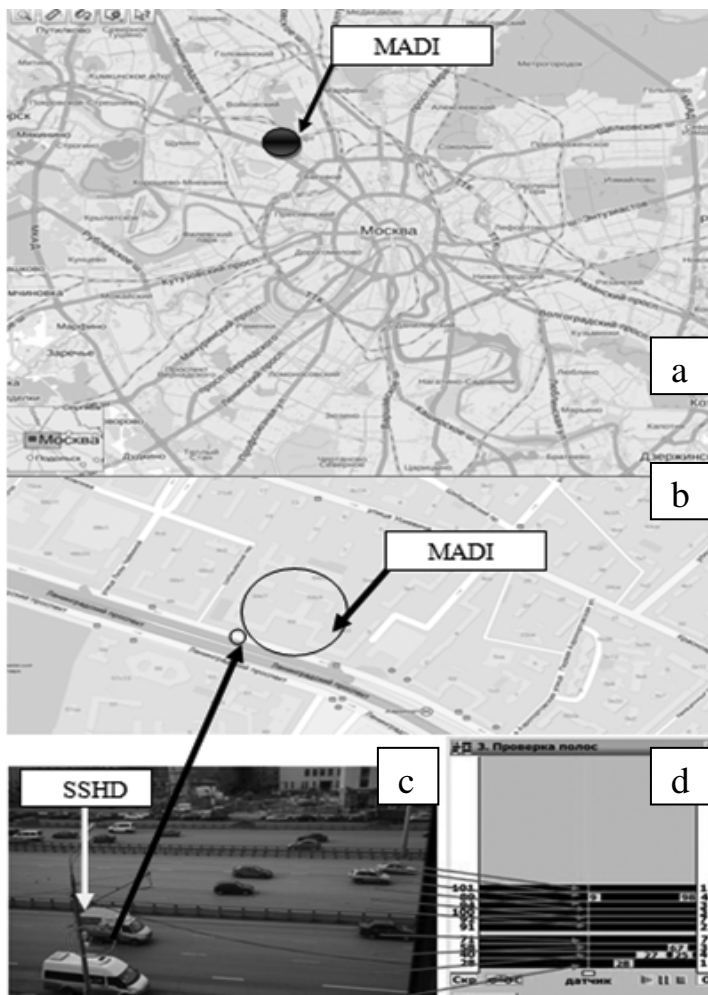


Fig. 1 a) Moscow map; b) MADI; c) SSHHD location; d) Corresponding lanes

2 Evaluation of accuracy for Smart Sensor HD

Let’s review the technical features given by the radar’s producer Wavetronix (www.wavetronix.com).

The device SSHD has the working frequency of 24-24.25 GCPS and allows to measure the quantity of passing vehicles, top fractile velocity, velocity, distance between the cars, direction of the flow, traffic density, intensity by lanes, defines the presence of cars and classifies them. The working zone of SSHD (coverage) is up to 80 meters and it allows managing and monitoring up to 10 lanes simultaneously. Numeration is set from the lane which is the closest to the installed radar (Fig. 1)

The producer states the following accuracy for the device and its measurements: deviation for intensity equals $\pm 10\%$, minimum distance to detect between 2 sequentially moving cars equals 1.67 m, deviation for velocity: ± 5 km/h.

Evaluation of SSHD accuracy was conducted via simultaneous video processing in order to define the limitations of this method for statistical data collection. The video was processed in laboratory conditions and with the help of the software which uses the automated processing of virtual detectors’ field [2]. The following values were found:

Table 1 Radar deviation

Number of lane	Deviation of intensity, %
1	*
2	57,5
3	60
4	42,5
5	30,6
6	30,3
7	35,8
8	33,1
9	37,4
10	24,3
Sum	39,1

* While defining deviation of intensity on the 1st lane we observed that the radar detected the intensity equal to 49 cars per 5 minutes even though there was a truck parked on this lane right at that time. The same mistake repeated several times.

In Table 1 the data is presented in the mode of wide moving jams (as per Kerner [8] definition). The deviation stated by the producer differs a lot from the deviation collected in the mode of wide moving jams. Therefore, the measurements with the SSHD radar are incorrect in the mode of wide moving jams and “stop-and-go”.

The main causes of registered deviations:

- Changing lanes by vehicles
- Overlapping of one car by another (shadow)
- Breaking the rules of radar’s installation
- The distance between detected cars is lower than critical (*1.7 m*)

3 Modes of Data Collection via SSHD

SSHD supports 2 modes of data collection:

- Interval (average for set time interval)

#	# NAME	VOLUME	Occu- pancy (%)	Speed (KPH)	85% Speed (KPH)	Class count (bin lengths in meters)								HEADWAY	GAP	SENSOR TIME YYYY-MM-DD HH:MM:SS
						C1	C2	C3	C4	C5	C6	C7	C8			
#						5,7	77,7									
	LANE_01	1	0,6	62,8	64,4	1	0	-	-	-	-	-	-	60,0	59,6	2011-07-03 12:15:00
	LANE_02	15	10,5	65,5	74,0	13	2	-	-	-	-	-	4,0	3,6	2011-07-03 12:15:00	

Fig. 2 Generic view of collected data in interval mode

- Individual (for each car)

#	LANE	LENGTH	(KPH)	CLASS	RANGE	SENSORTIME YYYY-MM-DD HH:MM:SS.sss	#
#							#
	LANE_09	5,3	98,0	1	26	2011-07-05 10:36:16.291	
	LANE_04	3,5	82,0	1	11	2011-07-05 10:36:17.203	

Fig. 3 Generic view of collected data in individual mode

In interval mode the radar is recording the features of the flow into text file on the set up averaging (For research accuracy the interval for data averaging is equal to 1 minute.)

In individual mode the radar is recording the features of cars passing the highway on-line into text file.

The significant drawback of the collected data is that it forms the tremendous array of information without graphical or table summary on the requested interval of time.

Also the “noises” (gaps) were detected in collected data, which happen because of the imperfect program support that is installed on SSHD. Thus, there is the missing or incorrectly registered data (empty lines, 100% intensity, negative velocity etc.).

In order to process the collected data (interval and individual), clean out the gaps and receive the features of the flow required by the scientists for the certain

periods of time, and also for the purpose of visualization, sorting and convenience to work with such massive information, the special software was found - «Software of data processing with SSM HD»

4 Description of the Software «The Program of Data Processing with SSM HD»

In order to process the collected data special software for SSM HD data was found. The software is written in the space Borland Delphi on the language Object Pascal. It serves the following functions:

- processes the collected data cleaning all the gaps and program mistakes;
- extracts density, velocity, intensity with set up averaging and building up graphs (dependence of values on time) and tables.

In the tables (Fig. 2-3) the columns describe time, the rows are number of the observed lane. The cells contain the values of hourly distribution for the required feature (intensity, velocity or density) of the traffic flow. The last columns and rows contain sums and percentage values. While building up graphs it is possible to choose any lane or/and summary value for the requested feature of traffic flow. Example of the table below shows the hourly distribution of intensity where the phases of the traffic flow can be revealed [10]:

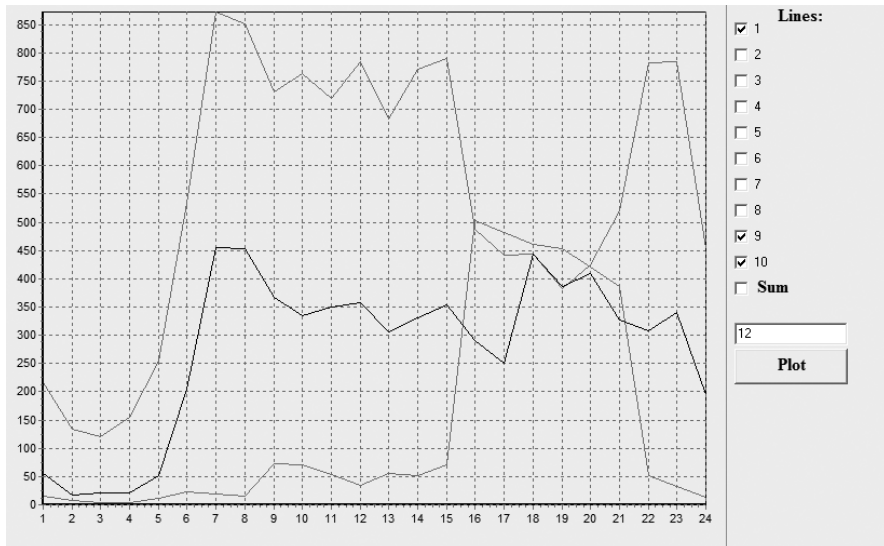


Fig. 4 Graph for hourly intensity on lanes -1,9 & 10 in the last 24 hours

Below there are the main features of traffic flow – density, velocity and intensity.

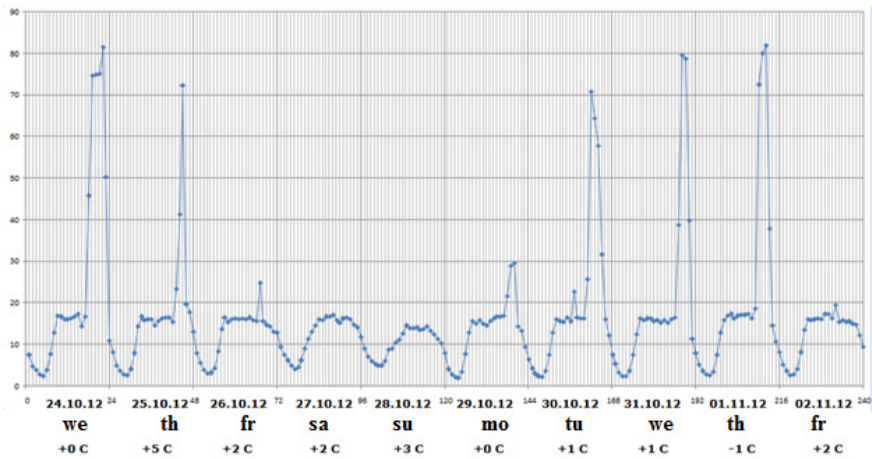


Fig. 5 Dynamic of density within 10 days

Analyzing the typical density graph it can be stated that the most intensive flow is on Wednesdays and Thursdays as there are peaks in these days. In other days we observe the even distribution of velocity.

It was noted, that average density within a year increased from 35,4 vehicles/km to 47,3 vehicles/km, in other words the average density increased 11,9 vehicles/km (33,6%)

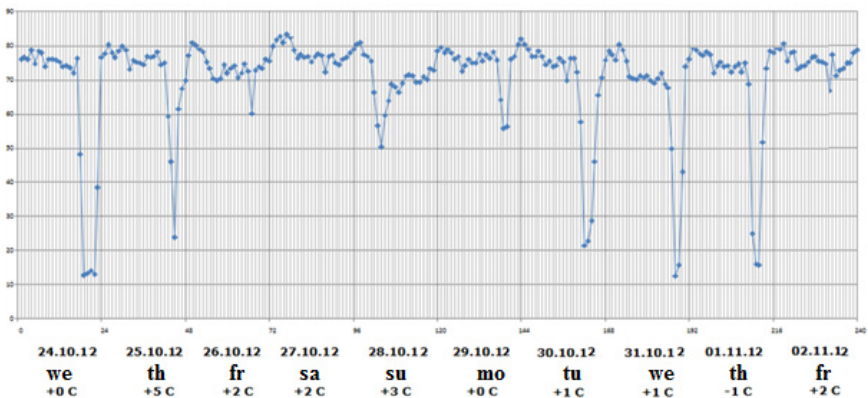


Fig. 6 Dynamic of average velocity within 10 days

As per Fig. 6 it is obvious that velocity has the areas of sharp decline on Wednesdays and Thursdays.

Average velocity within a year decreased from 64 km/h to 51 km/h, in other words the average velocity decreased 13 km/h. (20,3%)

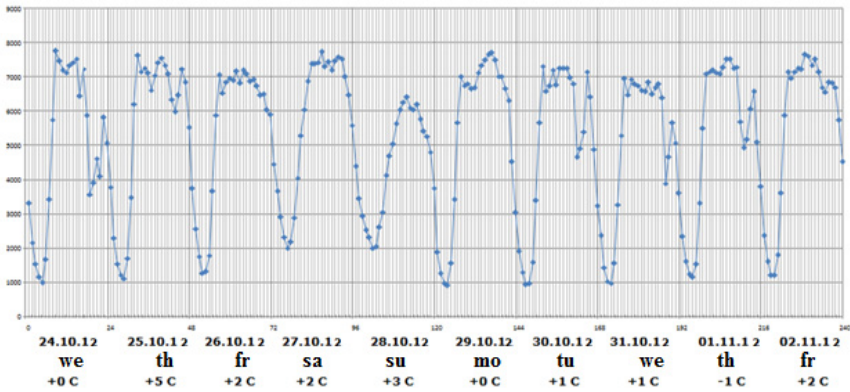


Fig. 7 Dynamic of intensity within 10 days

Analyzing the graphs the regularity of intensity’s sharp decline was revealed in midweek on Wednesdays and Thursdays.

Average intensity has decreased within a year from 45601 vehicles/h to 38851 vehicles/h, so it decreased on 6750 vehicles/h (14,8%).

It was noted that time in pre-traffic and traffic modes increased up to 4 hours which is on 50% more than the previous year. Based on the data it is obvious that the time period when the distance between cars is below 1.67m is not lower than 4 hours and is constantly going up. This means that the time for which SSHD gives the highest deviation (Table 1) is constantly increasing. At an instant within at least 4 hours a day (approximately 15%) the collected data is incorrect.

The periodic dependence of the form and area, evolved by traffic curves, from day of the week was found.

5 Problems of Traffic Flows in Moscow

Based on the data presented above it can be stated that the observed area by Leningradskoe highway 64 is capable to hold even higher intensity not leading to congestion but other road areas do not have that capacity and therefore it leads to congestion.

Let us review the main causes of traffic in Moscow:

I. Underdeveloped road networks

Based on Moscow Committee of Statistics the total length of Moscow streets, highways and driveways is equal to approximately 4500 km, in other words the quantity of roads per square km equals about 4,4 km. of roads/km².

II. Low junction

Lack of junctions in knots

III. Organization of road traffic

In Moscow 151 km of public transport lanes were allocated (around 5% from total length). This program was held in 2011-12, which led to capacity decrease of the road areas where there are no proper junctions.

IV. Violation of driving rules

This problem can be accredited to the distinctive features of national mentality (jagged movement, reckless driving).

Let's review the typical example of traffic development – area on Fig. 8.

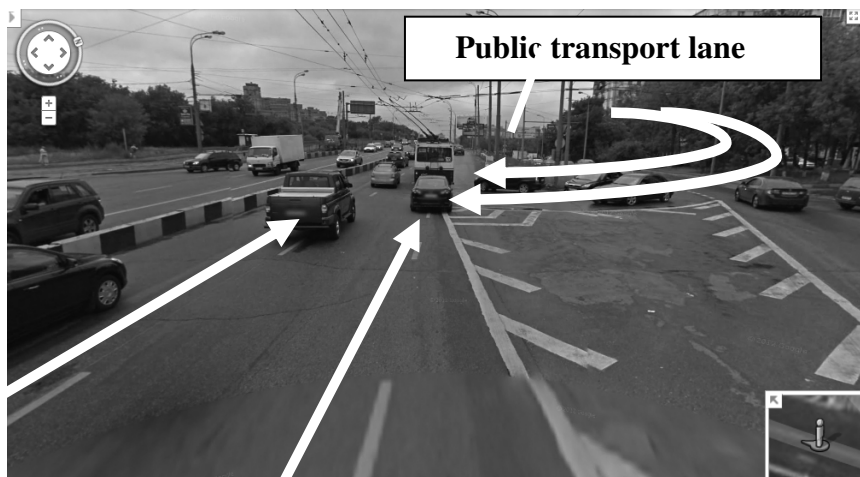


Fig. 8 Typical road network with 3 types presented

In Fig.8 there is an example of “bottleneck” caused by the reasons mentioned above. At such places intensive flows of cars come together. In the junction area the road has lower number of lanes and the total capacity is decreased due to allocated public transport lane (on Fig.8 the quantity of lanes decreased to 2), violation of driving rules (drivers turn not from one but 2 lanes as indicated in Fig.8) and etc. Such areas are obviously narrowed and this leads to density increase and velocity decrease.

Such areas, typical for Moscow, combining several junction problems, are the main reasons for traffic development. Traffic leads to high economic losses and their estimation is presented below.

6 Estimation of Economic Losses on the Most Typical Road of Moscow Traffic Network

Let us estimate economic losses on the typical tract of Moscow.

The drivers following to Volokolamskoe direction (Drivers VD) have to overcome approximately 2,3 km. The drivers following to Leningradskoe direction

(Drivers LD) have to overcome approximately 2,5 km. Average velocity of vehicles in free traffic is 75km/h. Quantity of vehicles passing to Volokolamskoe direction (VH) in stop-and-go mode is 4 380 vehicles. Quantity of vehicles passing to Leningradskoe direction (LH) is 20 380 vehicles. Time losses: VH & LH is 17 min per day (24h). Let us assume that average number of people in single vehicle is 1,3 people. Average salary per person is 290 RUB/h.

Thus, economic losses per day (24h) on road intervals №1,2,3,4 in direction to suburban areas each week day in high density equal: **3 000 000 RUB/24h**. Thus the driver who is passing the researched area is losing around 100 rub.

7 Summary

It has been shown that the method of data collection to measure traffic via microwave radar SSHD has satisfactory quality for 20-21 hours per day. Therefore, it requires alternative methods of automated monitoring.

The cause analysis of jam appearance was showed. Typical problems of road networks were revealed and estimation of economic losses for Moscow citizens is conducted.

Acknowledgements. The authors would like to express gratitude to their scientific supervisor A.P. Buslaev for the statement of problem and discussion of results.

This work has been supported by the Russian Foundation for Basic Research, grant No. 11-07-00622a.

References

- [1] Bugaev, A.S., Buslaev, A.P., Kozlov, V.V., Yashina, M.V.: Some mathematical and information aspects of traffic modeling. T-Comm - Telecommunications and Transport (4), 29–31 (2011)
- [2] Buslaev, A.P., Kuzmin, D.M., Yashina, M.V.: Computer methods of information processing and pattern recognition problems in transport and communications. Intelligent Systems in Transportation and Communication, vol. 3 (2008)
- [3] Buslaev, A.P., Gorodnichev, M.G.: Microwave Eye of "Big Brother": What is Visible from the Window of MADI. Ninth International Conference on Traffic and Granular Flow 2011, Book of abstracts. - M.T.-Comm, pp. 338–340 (2011)
- [4] Buslaev A.P., Gorodnichev M.G., Yashina M.V. Intellection systems SSHD: monitoring of multiband movement and automatic data processing of traffic. M. - МТУСИ, 100 стр (2012)
- [5] Daganzo, C.F.: Problem Sets: Fundamentals of Transportation and Traffic Operations, Institute of Transportation Studies, University of California at Berkley (1998)
- [6] Gorodnichev, M.G.: Some mathematical problems of car-following model. In: Proc. of Int. Conf. CMMSE 2012, vol. 2, pp. 673–677 (2012)
- [7] Greenshields, B.D.: The Photographic Method of Studing Traffic Behavior Highway RES. In: Board Proc., vol. 13 (1933)

- [8] Kerner, B.S.: Introduction to Modern Traffic Flow Theory and Control. Springer, Berlin (2009)
- [9] Kozlov, V.V., Buslaev, A.P.: Metropolis Traffic Modeling: From Intelligent Monitoring through Physical Representation to Mathematical Problems. In: Proc. of Int. Conf. CMMSE 2012, vol. 1, pp. 750–756 (2012)
- [10] Lighthill, M.J., Whitham, G.B.: On Kinematic Waves: Theory of Traffic Flow on Long Crowded Roads. Proc. R. Soc. London, Ser. A (1955)
- [11] Nagel, K., Schreckenberg, M.: A Cellular Automation Model For Freeway Traffic. Phys. I France 2 (1992)
- [12] Tihonov, A.V., Nigmatulin, A.N.: Technology of automatic capture of information on parameters of the distributed dynamic systems. Mobile laboratory of intellectual monitoring OTROK. Tehpoligraphcenter, 73 c (2012)

Integral Functionals of semi-Markov Processes in Reliability Problems

Franciszek Grabski

Naval University
Gdynia, Poland
F.Grabski@amw.gdynia.pl

Abstract. The possibility of calculating the reliability characteristics and parameters on the basis of stochastic process models of an object degradation is considered in the paper. Integral functionals of semi-Markov processes, called cumulative processes, have been used for construction stochastic models of degradation.

1 Introduction

The technical object wear is the effect of various damaging factors which have a random character. In many cases, the wear is the result of the cumulative effects of extortion. Damage of the object occurs when the cumulative effects of extortion exceed a certain level of degradation. Mathematical modelling of various forms of real objects wearing processes is a complex problem. Here we consider only cases, which lead to relatively simple models. Possibility of calculating the reliability characteristics and parameters on the basis of stochastic process models of an object degradation is considered in the paper. Integral functional of the semi-Markov processes have been used for construction the stochastic models of degradation.

2 Definition and Basic Properties of Integral Functional of the semi-Markov Process

Integral functional of the semi-Markov processes were presented by D. Silvestrov (1980), N. Limnios & G. Oprisan (2001). We present some concepts and theorems concerning such kind of the stochastic process that were discussed in [1], [2], [3]. We have to begin with a basic definition of the discrete states space semi-Markov processes theory.

Let $\mathbb{N}_0 = \{0, 1, 2, \dots\}$, $\mathbb{R}_+ = [0, \infty)$ and S denotes a discrete set. (Ω, \mathcal{F}, P) be a probability space. Suppose that $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$ is the sequence of random variables such that $\xi_n : \Omega \rightarrow S$ and $\vartheta_n : \Omega \rightarrow \mathbb{R}_+$. Two-dimensional sequence of the random variables $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$ is called a *Markov renewal process* (MRP) if

- for all $n \in \mathbb{N}_0, j \in S, t \in \mathbb{R}_+$

$$\begin{aligned} P(\xi_{n+1} = j, \vartheta_{n+1} \leq t \mid \xi_n = i, \vartheta_n, \dots, \xi_0, \vartheta_0) = \\ = P(\xi_{n+1} = j, \vartheta_{n+1} \leq t \mid \xi_n = i), \quad \text{with probability 1.} \end{aligned} \quad (1)$$

- for all $n \in \mathbb{N}_0, j \in S, t \in \mathbb{R}_+$

$$P(\xi_0 = i, \vartheta_0 = 0) = P(\xi_0 = i). \quad (2)$$

A function

$$Q_{ij}(t) = P(\xi_{n+1} = j, \vartheta_{n+1} \leq t \mid \xi_n = i) \quad (3)$$

is a transition probability of the two-dimensional $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$ Markov chain. We can see that the transition probability does not depend on a second (continuous) coordinate. A square matrix

$$\mathbf{Q}(t) = [Q_{ij}(t) : i, j \in S], t \geq 0 \quad (4)$$

is called a renewal kernel. A one row matrix

$$\mathbf{p}(0) = [P(\xi_0 = i) : i \in S] \quad (5)$$

is an initial distribution of the MRP. The renewal kernel is said to be *continuous* if at least its one row has an element containing an absolutely continuous component in Lebesgue decomposition.

Let

$$\begin{aligned} \tau_0 &= \vartheta_0, \\ \tau_n &= \vartheta_1 + \vartheta_2 + \dots + \vartheta_n, \quad n = 1, 2, \dots \\ \tau_\infty &= \lim_{n \rightarrow \infty} \tau_n = \sup\{\tau_n : n \in \mathbb{N}_0\} \end{aligned}$$

A stochastic process $\{N(t) : t \geq 0\}$ defined as

$$N(t) = n \quad \text{for} \quad \tau_n \leq t < \tau_{n+1}, \quad n \in \mathbb{N}_0 \quad (6)$$

is called a counting process of the MRP $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$.

A stochastic process $\{X(t) : t \geq 0\}$ given by a formula

$$X(t) = \xi_{N(t)} \quad (7)$$

is said to be a semi-Markov process generated by the MRP $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$ with the initial distribution $\mathbf{p}(0)$ and the kernel $\mathbf{Q}(t), t \geq 0$. From (6) and (7) we obtain

$$X(t) = \xi_n \quad \text{dla} \quad \tau_n \leq t < \tau_{n+1}, \quad n \in \mathbb{N}_0 \quad (8)$$

It follows from the above formula that the trajectories of SM process are the functions piecewise constant and right-continuous. SM process is called regular if for all $t \geq 0, P(N(t) < \infty) = 1$.

Suppose that $\{X(t) : t \geq 0\}$ is SM Markov regular process with continuous kernel. Assume the $h : S \rightarrow \mathbb{R}_+$ is any function taking values on a subset $U = h(S) \subset \mathbb{R}_+ = [0, \infty)$. Stochastic process

$$L(t) = \int_0^t h[X(u)]du \quad \text{with pr. 1} \tag{9}$$

is called an *integral functional* of the SM process or a *cumulative process* of the SM process $\{X(t) : t \geq 0\}$. If $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$ is the MRP defining the process $\{L(t) : t \geq 0\}$ then

$$L(t) = h(\xi_0) \vartheta_1 + \dots + h(\xi_{n-1}) \vartheta_n + h(\xi_n)(t - \tau_n) \quad \text{for } t \in [\tau_n, \tau_{n+1}) \tag{10}$$

This formula allows to generate the trajectories of the process $\{L(t) : t \geq 0\}$. Using the definition of the counting process we obtain an equivalent form of the formula (10)

$$L(t) = \sum_{k=1}^{N(t)} h(\xi_{k-1})\vartheta_k + (t - \tau_{N(t)})h(\xi_{N(t)}) \tag{11}$$

Example 1

Let $\{X(t) : t \geq 0\}$ be a semi-Markov process with a state space $S = \{0, 1, 2\}$ which is generated by MRP $\{(\xi_n, \vartheta_n) : n \in \mathbb{N}_0\}$. Assume $h(x) = 0,5x, \quad x \geq 0, \quad 0 \leq t \leq 12$ and

$$\begin{aligned} &\{(\xi_0 = 2, \vartheta_0 = 0), (\xi_1 = 1, \vartheta_1 = 4, 2), (\xi_2 = 0, \vartheta_2 = 2, 8), \\ &(\xi_3 = 2, \vartheta_3 = 1, 2), (\xi_4 = 2, \vartheta_4 = 3, 1), \dots\}. \end{aligned} \tag{12}$$

From (10) we obtain a piece of trajectory of the stochastic process $\{L(t) : t \geq 0\}$:

$$L(t) = \begin{cases} t & \text{for } t \in [0, 4.2) \\ 4.2 + 0,5(t - 4.2) & \text{for } t \in [4, 2, 7) \\ 5.6 & \text{for } t \in [7, 8.2) \\ 5.6 + 0.5(t - 8.2) & \text{for } t \in [8.2, 11.3) \\ 7.15 + (t - 11.3) & \text{for } t \in [11.3, \tau_5) \end{cases} \tag{13}$$

Figure 1 shows a trajectory of a semi-Markov process corresponding to a MRP realization (12) and figure 2 illustrates the corresponding trajectory of the integral functional.

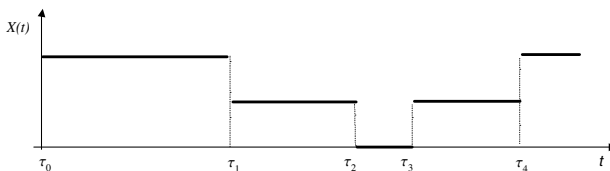


Fig. 1 Trajectory of a semi-Markov process

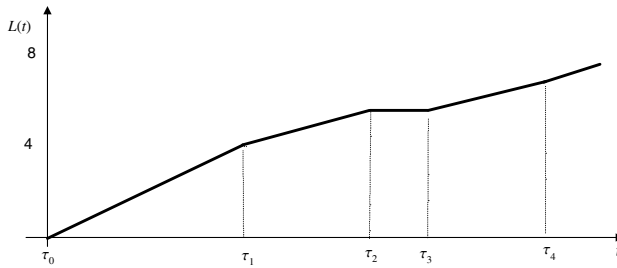


Fig. 2 A trajectory of the integral functional of semi-Markov process

Consider a joint distribution of processes $\{X(t) : t \geq 0\}$ and $\{L(t) : t \geq 0\}$ [3], [4]. Let

$$U_{iA}(t, x) = P(X(t) \in A, L(t) \leq x | X(0) = i), \quad i \in S. \tag{14}$$

The functions $U_{i,A}(t, x), i \in S$ satisfy a system of integral equations

$$U_{iA}(t, x) = I_{A \times [0, x]}(i, h(i) t)[1 - G_i(t)] + \sum_{j \in S} \int_0^t U_{jA}(t - v, x - h(i)v) dQ_{ij}(v), \quad i \in S \tag{15}$$

For $A = S$ we obtain

$$U_{iS}(t, x) = P\{L(t) \leq x | X(0) = i\}, \quad i \in S \tag{16}$$

The conditional cumulative distribution functions (CDF) of the process $\{L(t) : t \geq 0\}$ satisfy a system of the integral equations

$$U_{iS}(t, x) = I_{[0, x]}(h(i) t)[1 - G_i(t)] + \sum_{j \in S} \int_0^t U_{jS}(t - v, x - h(i)v) dQ_{ij}(v) \tag{17}$$

From (12) we get

$$U_{iA}(t, \infty) = \lim_{x \rightarrow \infty} U_{iA}(t, x) = P\{X(t) \in A | X(0) = i\}, \quad i \in S$$

Let

$$P_{iA}(t) = U_{iA}(t, \infty), \quad i \in S$$

The conditional probability $P_{iA}(t), i \in S$ verifies a system of integral equations

$$P_{iA}(t) = I_A(i)[1 - G_i(t)] + \sum_{j \in S} \int_0^t P_{jA}(t - v) dQ_{ij}(v), \quad i \in S \tag{18}$$

If A denotes subset of „up” states of the object , then $P_{iA}(t)$ denotes its *availability* under condition that an initial state is $i \in S$.

A cumulative process $\{L(t) : t \geq 0\}$ allows defining a random process $\{T(x) : x \geq 0\}$ by

$$T(x) = \inf\{t : L(t) > x\} \tag{19}$$

A random variable $T(x)$ denotes an instant of a level x exceeding by the cumulative process. If x is the critical level of the process describing degradation of an object then

$$R(t) = P(T(x) > t), \quad t \geq 0 \tag{20}$$

is a reliability function.

A stochastic process $\{K_j(t) : t \geq 0\}$ defined by

$$K_j(t) = \int_0^t I(X(u) = j) du, \tag{21}$$

where

$$I(X(u) = j) = \begin{cases} 1 & \text{dla } X(u) = j \\ 0 & \text{dla } X(u) \neq j \end{cases}, \tag{22}$$

is an example of an integral functional of a SM process. A value of the random variable $K_j(t)$ denotes a cumulated sojourn time of the SM process $\{X(t) : t \geq 0\}$ in a state j , during the interval $[0, t]$. The process $\{K_j(t) : t \geq 0\}$ is connected with the process $\{T_j(x) : x \geq 0\}$ defined by

$$T_j(x) = \inf\{t : K_j(t) > x\}. \tag{23}$$

A random process $\{T_j(x) : x \geq 0\}$ denotes an instant of time of exceeding a level x by the cumulated sojourn time of the SM process. Those processes have asymptotically normal distribution with parameters depending on a kernel of the process $\{X(t) : t \geq 0\}$. In renewal theory there is well known concept of an alternating process. We can treat it as the SM process $\{X(t) : t \geq 0\}$ with a state space $S = \{0, 1\}$, a kernel

$$Q(t) = \begin{bmatrix} 0 & F_\gamma(t) \\ F_\zeta(t) & 0 \end{bmatrix} \tag{24}$$

and an initial distribution

$$p(0) = [0, 1], \tag{25}$$

where $F_\gamma(t)$, $F_\zeta(t)$ are CDF of the nonnegative, independent random variables γ, ζ . From definition of the process $\{K_j(t) : t \geq 0\}$ it follows that the process is connected with the cumulated sojourn time in a state 1 of the alternating process [1], [2], [3]. If $\zeta_n, n = 1, 2, \dots$, denoting consecutive waiting

times of the state j , are supposed to be the random variables with an identical probability

$$G_j(t) = P(\zeta \leq t) = P(T_j \leq t) \tag{26}$$

while $\gamma_n, n = 1, 2, \dots$ denote the lengths of the time intervals, that pass from the instants of n -th going out from the state j to next going in this state, then the definition of the process $\{K_j(t) : t \geq 0\}$ which start with the state j is almost identical with definition of the simple alternating process. The only difference lies in that for general SM process, the random variables $\zeta_n, \gamma_n, n = 1, 2, \dots$ can be dependent. But we can distinguish a class of the SM processes, that the mentioned above random variables are independent.

If $\{X(t) : t \geq 0\}$ is SM process with a kernel $Q(t) = [Q_{ij}(t) : i, j \in S]$ such that $Q_{ij}(t) = p_{ij}G_j(t)$, where $p_{ii} = 0$ for $i \in S$, then the random variables $\zeta_n, \gamma_n, n = 1, 2, \dots$ are independent and

$$\Theta_{jj}^{(n)} = \zeta_n + \gamma_n \tag{27}$$

where $\Theta_{jj}^{(n)}$ is n -th return time to the j state [3].

The equations which allow to calculating the one dimensional distribution of the process $\{K_j(t) : t \geq 0\}$ are presented in books [2], [3]. But they are very complicated and difficult for applying. Then, in this case we can use an approximate formula which implies from the following theorem [2], [3]:

Let $\{X(t) : t \geq 0\}$ be a SM process defined by a continuous type kernel $Q(t) = [Q_{ij}(t) : i, j \in S]$ such that

$$Q_{ij}(t) = p_{ij}G_j(t), \quad \text{gdzie } p_{ii} = 0 \quad \text{dla } i \in S$$

If moreover the random variables T_j, Θ_{jj} have positive finite second moments, then

$$\lim_{t \rightarrow \infty} P \left(\frac{K_j(t) - m_j(t)}{\sigma_j(t)} \leq x \right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{u^2}{2}} du \tag{28}$$

where

$$m_j(t) = \frac{E(T_j)}{E(\Theta_{jj})} t \tag{29}$$

$$\sigma_j(t) = \sqrt{\frac{V(T_j)[E(\Theta_{jj}) - E(T_j)]^2 + [V(\Theta_{jj}) - E(T_j)][E(T_j)]^2}{[E(\Theta_{jj})]^3}} t \tag{30}$$

From the above theorem it follows that the process $\{K_j(t) : t \geq 0\}$ has an approximately normal distribution with an expectation given by an equality (22) and a standard deviation taking the form of (30).

Under the same assumptions the process $\{T_j(x) : x \geq 0\}$ denoting the moment of exceeding a level x by the a cumulated sojourn time of the SM process has an approximately normal distribution

$$\lim_{t \rightarrow \infty} P\left(\frac{T_j(x) - m_j(x)}{\sigma_j(x)} \leq y\right) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^y e^{-\frac{u^2}{2}} du \tag{31}$$

where

$$m_j(x) = \frac{E(\Theta_{jj})}{E(T_j)} x, \tag{32}$$

$$\sigma_j(x) = \sqrt{\frac{V(T_j) [E(\Theta_{jj}) - E(T_j)]^2 + [V(\Theta_{jj}) - V(T_j)] [E(T_j)]^2}{[E(T_j)]^3}} x. \tag{33}$$

For the alternating process $\{X(t) : t \geq 0\}$ with the kernel (24) the formulas (29), (30), (32), (33) are of the form

$$m(t) = \frac{E(\zeta)}{E(\zeta) + E(\gamma)} t, \tag{34}$$

$$\sigma(t) = \sqrt{\frac{V(\zeta) [E(\gamma)]^2 + [V(\gamma)] [E(\zeta)]^2}{[E(\zeta) + E(\gamma)]^3}} t, \tag{35}$$

$$m(x) = \frac{E(\zeta) + E(\gamma)}{E(\zeta)} x, \tag{36}$$

$$\sigma(x) = \sqrt{\frac{V(\zeta) [E(\gamma)]^2 + [V(\gamma)] [E(\zeta)]^2}{[E(\zeta)]^3}} x. \tag{37}$$

3 Example

After $c = 50000$ [h] of working a plane engine is treated as broken-down. We suppose that a sojourn time of one fly is random variable ζ with an expectation $E(\zeta) = 3.48$ [h] and a variance $V(\zeta) = 2.15$ [h²]. A time of the each plane stoppage is a positive random variable γ with the expectation $E(\gamma) = 4.5$ [h] and the variance $V(\gamma) = 4.21$ [h²]. Under those assumption the alternating process $\{X(t) : t \geq 0\}$ defined by the kernel (24) and the initial distribution (25) is a reliability model of the plane engine operation process. A random variable $T(c) = \inf\{t : K(t) > c\}$, where $K(t) = \int_0^t X(u) du$ denotes an instant of exceeding a level c by a summary sojourn time of the alternating process $\{X(t) : t \geq 0\}$. From above presented theorem it follows that a random variable $T(c)$ has an approximately normal distribution $N(m(c), \sigma(c))$, where

$$m(c) = \frac{E(\zeta) + E(\gamma)}{E(\zeta)} c$$

and

$$\sigma(c) = \sqrt{\frac{V(\zeta) [E(\gamma)]^2 + [V(\gamma)] [E(\zeta)]^2}{[E(\zeta)]^3}} c.$$

The estimated reliability function $R(t)$, $t \geq 0$ takes the form:

$$R(t) = P(T(c) > t) \approx 1 - \Phi\left(\frac{t - m(c)}{\sigma(c)}\right)$$

where $\Phi(\cdot)$ is CDF of the standard normal distribution. For above assumed parameters we obtain the expectation and the standard deviation for a time to failure of the plane engine.

$$m(50000) \approx 114655 [h] \approx 13,09 [lat], \quad \sigma(50000) \approx 334.0 [h] \approx 0.038 [lat]$$

The estimated reliability function $R(t)$, $t \geq 0$ is

$$R(t) \approx 1 - \Phi\left(\frac{t - 114655, 0}{334.0}\right)$$

The function is shown in figure 3. The value of the function for $c=50000$ is 0.99865.

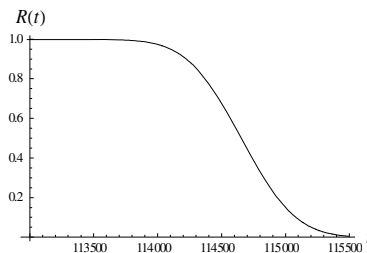


Fig. 3 The approximate reliability function of the plain engine

4 Conclusions

Cumulative processes can be used as a probability models of the object degradation. Theoretical results give possibility to obtain approximate reliability parameters and characteristics.

References

- [1] Grabski, F.: Theory of semi-Markov processes of the operation. *Zeszyty naukowe Marynarki Wojennej* 75A, s. 253 (1982)
- [2] Grabski, F.: Semi-Markov model of reliability and operation. In: *Operation Research*, vol. 30, p. 161. PAN IBS, Warszawa (2002) (in Polish)
- [3] Grabski, F., Jaźwiński, J.: Functions of random arguments applications in reliability, safety and logistics, p. 344. WKŁ, Warszawa (2009)
- [4] Limnios, N., Oprisan, G.: *Semi-Markov Processes and Reliability*. Birkhauser, Boston (2001)
- [5] Silvestrov, D.S.: *Semi-Markov processes with discrete state space*. Sovietskoe Radio, Moskva (1980) (in Russian)

Generating Repair Rules for Database Integrity Maintenance

Feras Hanandeh and Yaser Quasmeh

Prince Al-Hussein bin Abdullah II Faculty of Information Technology

Hashemite University, Jordan

feras@hu.edu.jo, Y_quasmeh@yahoo.com

Abstract. Repair system has two essential components, which are much related to each other. When the update operation is executed, the first component is the detection of the erroneous state if any and the second component is to repair this state by finding the changes to the update operation that would repair it. Failing to have the second component, which is the repair action, will enforce the user to manually correcting and reentering an erroneous update operation. Our approach will take advantage of the integrity before the update operation, which will result on limiting the detection only to the database state after the update operation. Also the repair component will take advantage of the integrity before the update operation and integrity violation after the update operation but before the repair. The focus of this paper is to generate repairs for all first order constraints, and by using only substitution with no resolution search. Multiple constraints can be satisfied in parallel without a sequential process with no possibility of cyclic violation.

1 Introduction

The reliability of information systems is a major concern for today's society and enterprises. The correctness or maintaining database integrity of databases is one of the main reliability issues. Consequently procedures asserting correct databases are a chief focus of research. Today the prime obstacles applying these procedures are their high computational costs. Integrity maintenance is considered one of the major application fields of rule triggering systems. In the case of a given integrity constraint being violated by a database transition these systems trigger update operation (action) to maintain database integrity.

A relational database is a collection of relations; each relation corresponds to a database predicate. Each relation R is a collection of tuples. Any attempt to update the database should be controlled by integrity constraints. When any of these constraints is violated by an update operation then the system should either abort or take action to repair the erroneous update operation. Such system is called integrity maintenance subsystem. When detecting any erroneous update operation, repairing become essential since detection without repairing the erroneous state

will never accommodate users need to guarantee consistency, accuracy and the integrity of their systems. The integrity maintenance subsystem separate the database state into two states, the first is before the update operation. The second state is after the update operation, so the integrity maintenance subsystem has to detect any new errors introduced by the update operation and if there is any error to be repaired. Our approach involves algebraically modifying the constraint definitions into derivative expressions that return the condition for a new violation to occur. The derived predicate is a predicate defined in terms of the database predicates. The derived predicate, which denotes a violation of database constraint, is considered as a negation of the constraint.

2 Related Work

Relational Databases usually contain massive collections of data that rapidly evolve over time; this makes perfect checking at each update too time consuming a task to be feasible. In this regard, DBMS needs to be extended with the ability to automatically verify that database updates do not introduce any violation of integrity [3; 11].

The primary tool of integrity maintenance subsystem is the database integrity rules. The aim of integrity rules is to capture the semantics of data. Integrity rules provide a much more general capability to maintain integrity than the data models since they can utilize the full power of logic based language. The high cost results from using integrity rules may become as a restriction since they often involve the execution of complex queries against a large database.

Automation of the various repairable systems was the main aim for the researchers in the last decade. Partial automation was the aim of some researchers like [1, 4, 5, 16 and 18]. They adopted the notion of entrust the final repair to be manually designed by the users provided that the guidelines which they have to follow for the repair operation is clearly generated. Other approaches [14, 17 and 19] generated sufficient conditions for repair by the user entrusting to him the final repair to be manually designed by pruned the necessary repair, a suitable decision making framework based on encompassing all the actions requested to repair the erroneous state formulated, since there is not minimal repair actions. Some approaches resort to impose severe restrictions on the quantifier structure of the constraints like no existential quantifiers followed by universal quantifiers [1, 3 and 4].

Expensive rollback is the repair action adopted by many approaches [1, 4, 5, 14, 16 and 19] since executing of the update operation first was the condition for checking any possible integrity violations.

Soumya et al [15] proposed a technique to achieve optimization of constraint checking process in distributed databases by exploiting technique of parallelism, compile time constraint checking, localized constraint checking, and history of constraint violations. The architecture mainly consists of two modules: Constraint Analyzer and Constraint Ranker for analyzing the constraints and for ranking the

constraints, respectively for systems with relational databases. They achieved optimization in terms of time by executing the constraints in parallel with mobile agents.

3 Preliminaries

Our approach has been developed in the context of relational databases. A relational database is a collection of relations, each corresponding to a database predicate. Each relation P is a collection of tuples T_i satisfying the corresponding predicate P , i.e., $P(T_i)$ is True. An intentional (derived) predicate is a predicate defined in terms of the database predicates. Let V be an intentional predicate that denotes a violation of the database integrity constraint C , (i.e., V is the negation of C), where an integrity constraint is the primary tool of integrity maintenance system specifying a condition that should be satisfied by the database. Efficient computation of V is critical in detecting semantic violations caused by erroneous database update operations. A database update transaction is defined as a collection of insertions into and deletions from the database.

Throughout this paper the same example Job Agency database is used, as given below. This example is taken from [19]:

Person (pid, pname, placed)
 Company (cid, cname, totals)
 Job (jid, jdescr)
 Placement (pid, cid, jid, sal)
 Application (pid, jid)
 Offering (cid, jid, no_of_places)

Definition 1. Given an update operation, for each database relation P , the incessant of the relation P , ΓP means that the tuples exists in relation P and not being deleted from relation P , such that:

$$\forall x (\Gamma P(x) \leftarrow P(x) \wedge \neg \delta P(x))$$

where δ is a deletion operator.

Definition 2. For every database relation P there are three different database states of the same relation, where P is the state of the relation before an update operation is performed, P' is the state of P after the update operation is performed and P'' is the state of P after the repair operation is executed.

Definition 3. Given an update operation, for each database relation P , the new database state of the relation P' means that the tuples incessant in the database relation P , i.e. ΓP and the tuples to be inserted by the update operation, such that:

$$\forall x (P'(x) \leftarrow \Gamma P(x) \vee \iota P(x))$$

Assumption 1. Given an update operation, for each database relation P , it is assumed that:

- $\neg\iota P$ means that a tuple(s) were deleted from P, i.e. δP .
- $\neg\delta P$ means that a tuple(s) were inserted into P, i.e. ιP .

where ι is an insertion operator.

Given the fact that a pre-transaction state of a database is correct, a reduction of the amount of data to be checked in constraint enforcement can be obtained by inspecting only those parts of relations that have been changed in a relevant way by a transaction. This is usually accomplished by the use of differential relations [6, 8, 9, 12 and 13]. In this approach, two auxiliary relations are associated with each base relation P.

New differential relation. The new differential relation contains the set of tuples to be inserted into the relation P or the new modified tuples by the current update operation. The new differential relation associated with relation P is denoted as ιP and can be defined as:

$\forall x (\iota P(x) \leftarrow \neg P(x) \wedge P'(x))$, i.e. ιP means that the tuples that is not in the relation P, but in P'.

Old Differential Relation. The old differential relation contains the set of tuples to be deleted from the relation P or the old tuples that have been modified by the current update operation. The old differential relation associated with relation P is denoted as δP and can be defined as:

$\forall x (\delta P(x) \leftarrow P(x) \wedge \neg P'(x))$, i.e. δP means that the tuples that is in the relation P, but not in P'.

Using this auxiliary relations, constraint conditions can be reformulated.

Example. Given the domain integrity constraint C that states the placed attribute in relation Person must be either T or F

$$(\forall v, \forall w, \forall x) \text{Person}(v, w, x) \Rightarrow x = T \vee x = F$$

Given the fact that the constraint holds on the pre-transaction state of relation Person, only the new tuples to be inserted into Person have to be checked. Therefore, the constraint can be optimized as follows:

$$(\forall v, \forall w, \forall x) \iota \text{Person}(v, w, x) \Rightarrow x = T \vee x = F$$

4 Generating Integrity Tests

The main aim of this research is to contribute to the solution of constraint checking in a parallel database by deriving integrity tests from a given constraint and a given update operation. Using integrity tests to detect violations instead of integrity constraints is more fruitful because the later is costly since they often involve the execution of complex queries against a large database. These integrity tests may be necessary, sufficient or complete.

This section presents the algorithm which can be employed to derive more integrity tests than the previous approaches discussed in the related work section. The algorithm should be as general as possible, i.e. independent of any specific application domain. We take out the quantifiers Q for simplicity.

Lemma 1. Given a predicate $P(X)$ in the form: $P(X) \leftarrow Q(Y) \wedge R(Z)$, $\delta P(X)$ will be computed as follows: $\delta P(X) \leftarrow (\delta Q(Y) \wedge R(Z)) \vee (\delta R(Z) \wedge Q(Y))$ i.e. either deleting a tuple(s) from the relation R or a tuple(s) from the relation Q will result in deleting a tuple(s) from the relation P . Deleting a tuples from both relations Q and R can be realized from the formula.

Proof

$$\begin{aligned} \delta P(X) &\leftarrow P(X) \wedge \neg P'(X) \\ \delta P(X) &\leftarrow Q(Y) \wedge R(Z) \wedge \neg(Q'(Y) \wedge R'(Z)) \\ \delta P(X) &\leftarrow Q(Y) \wedge R(Z) \wedge (\neg Q'(Y) \vee \neg R'(Z)) \\ \delta P(X) &\leftarrow (\delta Q(Y) \wedge R(Z)) \vee (\delta R(Z) \wedge Q(Y)) \end{aligned}$$

Lemma 2. Given a predicate $P(X)$ in the form: $P(X) \leftarrow Q(Y) \vee R(Z)$, $\delta P(X)$ will be computed as follows: $\delta P(X) \leftarrow (\neg \Gamma Q(Y) \wedge \delta R(Z)) \vee (\neg \Gamma R(Z) \wedge \delta Q(Y))$ i.e. not existence a tuple(s) of both relations R and Q will result in deleting a tuple(s) from the relation P .

Proof

$$\begin{aligned} \delta P(X) &\leftarrow P(X) \wedge \neg P'(X) \\ P(X) &\leftarrow (Q(Y) \vee R(Z)) \wedge \neg(Q'(Y) \vee R'(Z)) \\ &\leftarrow (Q(Y) \vee R(Z)) \wedge (\neg Q'(Y) \wedge \neg R'(Z)) \\ &\leftarrow (Q(Y) \wedge \neg Q'(Y) \wedge \neg R'(Z)) \vee (R(Z) \wedge \neg Q'(Y) \wedge \neg R'(Z)) \\ &\leftarrow (\delta Q(Y) \wedge \neg(R(Z) \wedge \neg \delta R(Z) \vee \neg R(Z))) \vee (\delta R(Z) \wedge \neg(Q(Y) \wedge \neg \delta Q(Y) \vee \neg Q(Y))) \\ &\leftarrow (\delta Q(Y) \wedge (\neg R(Z) \vee \delta R(Z)) \wedge \neg \neg R(Z)) \vee (\delta R(Z) \wedge (\neg Q(Y) \vee \delta Q(Y) \wedge \neg \neg Q(Y))) \\ &\leftarrow (\delta Q(Y) \wedge (\neg R(Z) \wedge \neg \neg R(Z) \vee \delta R(Z) \wedge \neg \neg R(Z))) \vee (\delta R(Z) \wedge (\neg Q(Y) \wedge \neg \neg Q(Y) \vee \delta Q(Y) \wedge \neg \neg Q(Y))) \\ &\leftarrow (\neg R(Z) \wedge \neg \neg R(Z) \wedge \delta Q(Y)) \vee (\delta R(Z) \wedge \neg \neg R(Z) \wedge \delta Q(Y)) \vee (\neg Q(Y) \wedge \neg \neg Q(Y) \wedge \delta R(Z)) \vee (\delta Q(Y) \wedge \neg \neg Q(Y) \wedge \delta R(Z)) \\ &\leftarrow (\neg R(Z) \wedge \neg \neg R(Z) \wedge \delta Q(Y)) \vee (\delta R(Z) \wedge \delta R(Z) \wedge \delta Q(Y)) \vee (\neg Q(Y) \wedge \neg \neg Q(Y) \wedge \delta R(Z)) \vee (\delta Q(Y) \wedge \delta Q(Y) \wedge \delta R(Z)) \\ &\leftarrow (\neg R(Z) \wedge \neg \neg R(Z) \wedge \delta Q(Y)) \vee (\delta R(Z) \wedge \delta Q(Y)) \vee (\neg Q(Y) \wedge \neg \neg Q(Y) \wedge \delta R(Z)) \vee \delta Q(Y) \wedge \delta R(Z) \\ &\leftarrow \neg(R(Z) \vee \neg \delta R(Z)) \wedge \delta Q(Y) \vee (\neg Q(Y) \wedge \neg \neg Q(Y) \wedge \delta R(Z)) \vee \delta Q(Y) \wedge \delta R(Z) \end{aligned}$$

The process of generate integrity tests starts by accepting an integrity constraint from the constraint base and a given update operation by the user. Figures 1 and 2 present the algorithms to generate integrity tests for constraints in conjunctive and disjunctive normal form respectively. Every generated integrity test is distributed to all dynamic virtual partitions [7] through generate distributed integrity test

algorithm. This algorithm not presented in this paper because of space constraint. A clarified example to illustrate our technique followed the algorithms.

Checking the validity of the integrity tests against the database is activated at run-time once these tests are generated (computed) by logical rules at compile-time. These logical rules are specified using logical specification language. Each generation rule (conjunctive or disjunctive) has three input expressions. The output of the generation rule is sufficient, necessary or complete test(s).

```

Algorithm (Generate Texp for a constraint in a conjunctive normal form)
Inputs: Constraint C in conjunctive normal form /* $\forall \leftarrow P(X) \wedge Q(Y)$ */
Update Operation (UO)
Output: Texp /*Integrity Test Expression*/
Method:
Begin
Step 1:
Exp1  $\leftarrow \neg P(X) \wedge \neg Q(Y)$  /*inessant of P(X) and insert a tuple into base relation Q */
where:  $\Gamma P \leftarrow P \wedge \neg \delta P$ 
Exp2  $\leftarrow \neg Q(Y) \wedge \neg P(X)$  /*inessant of Q(Y) and insert a tuple into base relation P */
where:  $\Gamma Q \leftarrow Q \wedge \neg \delta Q$ 
Exp3  $\leftarrow \neg P(X) \wedge \neg Q(Y)$  /*insert two tuples into both base relations P and Q */
Step 2:
Texp  $\leftarrow \text{Exp1} \vee \text{Exp2} \vee \text{Exp3}$ 
Step 3:
Negate Texp
End.

```

Fig. 1 Generate Texp algorithm for a constraint in a conjunctive normal form

```

Algorithm (Generate Texp for a constraint in a disjunctive normal form)
Inputs: Constraint IC in disjunctive normal form /* $\forall \leftarrow P(X) \vee Q(Y)$ */
Update Operation (UO)
Output: Texp /*Integrity Test Expression*/
Method:
Begin
Step 1:
Exp1  $\leftarrow \neg P(X) \wedge \neg Q(Y)$  /*insert a tuple into the base relation P and Q(Y) is false*/
Exp2  $\leftarrow \neg P(X) \wedge \neg Q(Y)$  /* P(X) is false and insert a tuple into the base relation Q */
Exp3  $\leftarrow \neg P(X) \wedge \neg Q(Y)$  /*insert two tuples into both base relations P and Q */
Step 2:
Texp  $\leftarrow \text{Exp1} \vee \text{Exp2} \vee \text{Exp3}$ 
Step 3:
Negate Texp
End.

```

Fig. 2 Generate Texp algorithm for a constraint in a disjunctive normal form

We now present detailed example to generate Texp algorithm which will clarify the steps presented above.

Example: when a company offers a Director job it must offer a Senior Secretary job first.

$$C1 \leftarrow (\forall x \forall y \forall z)(\text{Offering}(x, y, z) \wedge \text{Job}(y, \text{'Director'}) \wedge \neg S(x))$$

where $S(x) \leftarrow \text{Offering}(x, p, q) \wedge \text{Job}(p, \text{'Senior Secretary'})$

Update: $\text{Insert}(\text{Offering}(x, y, z)) = \{x/C, y/J, z/N\}$

$$C1 \leftarrow (\forall x \forall y \forall z)(\text{Offering}(x, y, z) \wedge M(x, y))$$

where $M(x, y) \leftarrow \text{Job}(y, \text{Director}) \wedge \neg S(x)$

Step1

Exp1 (Texp) $\leftarrow \Gamma \text{Offering}(x, y, z) \wedge \iota M(x, y)$

Exp2 (Texp) $\leftarrow \Gamma M(x, y) \wedge \iota \text{Offering}(x, y, z)$

Exp3 (Texp) $\leftarrow \iota \text{Offering}(x, y, z) \wedge \iota M(x, y)$

Computing $\delta S(x)$ using Lemma1

$S(x) \leftarrow \text{Offering}(x, p, q) \wedge \text{Job}(p, \text{'Senior Secretary'})$
 Exp1 $\leftarrow \delta \text{Offering}(x, p, q) \wedge \text{Job}(p, \text{'Senior Secretary'})$
 $\leftarrow \text{False} \wedge \text{Job}(p, \text{'Senior Secretary'})$
 $\leftarrow \text{False}$
 Exp2 $\leftarrow \text{Offering}(x, p, q) \wedge \delta \text{Job}(p, \text{'Senior Secretary'})$
 $\leftarrow \text{Offering}(x, p, q) \wedge \text{False}$
 $\leftarrow \text{False}$
 $\delta S(x) \leftarrow \text{Exp1} \vee \text{Exp2}$
 $\delta S(x) \leftarrow \text{False} \vee \text{False}$
 $\delta S(x) \leftarrow \text{False}$

Computing $\delta M(x, y)$ using Lemma1

$M(x, y) \leftarrow \text{Job}(y, \text{'Director'}) \wedge \neg S(x)$
 Exp1 $\leftarrow \text{Job}(y, \text{'Director'}) \wedge \iota S(x)$
 Exp2 $\leftarrow \neg S(x) \wedge \delta \text{Job}(y, \text{'Director'})$
 $\delta M(x, y) \leftarrow \text{Exp1} \vee \text{Exp2}$
 $\delta M(x, y) \leftarrow \text{Job}(y, \text{'Director'}) \wedge \iota S(x) \vee \text{False}$
 $\delta M(x, y) \leftarrow \text{Job}(y, \text{'Director'}) \wedge \iota S(x)$

Computing $\iota M(x, y)$

$M(x, y) \leftarrow \text{Job}(y, \text{'Director'}) \wedge \neg S(x)$
 Exp1 $\leftarrow \Gamma \text{Job}(y, \text{Director}) \wedge \neg \iota S(x)$
 $\leftarrow \Gamma \text{Job}(y, \text{Director}) \wedge \delta S(x)$
 $\leftarrow \Gamma \text{Job}(y, \text{Director}) \wedge \text{False}$
 $\leftarrow \text{False}$
 Exp2 $\leftarrow \neg \Gamma S(x) \wedge \iota \text{Job}(y, \text{'Director'})$
 $\leftarrow \neg(S(x) \wedge \neg \delta S(x)) \wedge \iota \text{Job}(y, \text{'Director'})$
 $\leftarrow (\neg S(x) \vee \delta S(x)) \wedge \iota \text{Job}(y, \text{'Director'})$
 $\leftarrow (\neg S(x) \vee \delta S(x)) \wedge \text{False}$
 $\leftarrow \text{False}$
 Exp3 $\leftarrow \iota \text{Job}(y, \text{'Director'}) \wedge \delta S(x)$
 $\leftarrow \text{False} \wedge \delta S(x)$
 $\leftarrow \text{False}$
 $\iota M(x, y) \leftarrow \text{Exp1} \vee \text{Exp2} \vee \text{Exp3}$
 $\iota M(x, y) \leftarrow \text{False} \vee \text{False} \vee \text{False}$
 $\iota M(x, y) \leftarrow \text{False}$

Computing $\iota S(x)$

$$\begin{aligned}
S(x) &\leftarrow \text{Offering}(x, p, q) \wedge \text{Job}(p, \text{'Senior Secretary'}) \\
\text{Exp1} &\leftarrow \Gamma \text{Offering}(x, p, q) \wedge \iota \text{Job}(p, \text{'Senior Secretary'}) \\
\text{Exp2} &\leftarrow \Gamma \text{Job}(p, \text{'Senior Secretary'}) \wedge \iota \text{Offering}(x, p, q) \\
\text{Exp3} &\leftarrow \iota \text{Offering}(x, p, q) \wedge \iota \text{Job}(p, \text{'Senior Secretary'}) \\
\iota S(x) &\leftarrow \text{Exp1} \vee \text{Exp2} \vee \text{Exp3} \\
\iota S(x) &\leftarrow \text{False} \vee \text{Job}(J, \text{'Senior Secretary'}) \wedge x = C \vee \text{False} \\
\iota S(x) &\leftarrow \text{Job}(J, \text{'Senior Secretary'}) \wedge x = C
\end{aligned}$$

Substituting the result of $\iota S(x)$ into $\delta M(x, y)$:

$$\delta M(x, y) \leftarrow \text{Job}(y, \text{'Directory'}) \wedge \text{Job}(J, \text{'Senior Secretary'}) \wedge x = C$$

Substituting the results of $\iota M(x, y)$ into Exp1 (Texp) and Exp3 (Texp):

$$\text{Exp1 (Texp)} \leftarrow \text{False}$$

$$\text{Exp3 (Texp)} \leftarrow \text{False}$$

$$\text{Exp2 (Texp)} \leftarrow \Gamma M(x, y) \wedge \iota \text{Offering}(x, y, z)$$

$$\leftarrow M(x, y) \wedge \neg \delta M(x, y) \wedge \iota \text{Offering}(x, y, z)$$

$$\leftarrow \text{Job}(y, \text{'Director'}) \wedge \neg S(x) \wedge \neg(\text{Job}(y, \text{'Director'}) \wedge \text{Job}(J, \text{'Senior Secretary'}) \wedge x = C) \wedge (x, y, z) = (C, J, N)$$

Step2

$$\text{Texp} \leftarrow \text{Job}(J, \text{'Director'}) \wedge \neg S(C) \wedge (\neg \text{Job}(J, \text{'Director'}) \vee \neg \text{Job}(J, \text{'Senior Secretary'})) \vee \neg(C = C)$$

By substitution and negation rule

$$\text{Texp} \leftarrow \text{Job}(J, \text{'Director'}) \wedge \neg S(C) \wedge \neg \text{Job}(J, \text{'Senior Secretary'})$$

Step3

$$\neg \text{Texp} \leftarrow \neg \text{Job}(J, \text{'Director'}) \vee S(C) \vee \text{Job}(J, \text{'Senior Secretary'})$$

where

$$S(C) \leftarrow \text{Offering}(C, p, q) \wedge \text{Job}(p, \text{'Senior Secretary'})$$

A number of sufficient tests can be computed by applying the substitution and resolution rules[2] to the sufficient and complete integrity tests in $\neg \text{Texp}$ and the original constraint C1. These sufficient tests are often easier to test than the complete tests, and only one of them needs to be tested to prove that there are no violations. Given integrity constraint C, let negation of its violation be $\neg \text{Texp} \leftarrow \text{SCT}$, where SCT is the sufficient and complete tests for integrity checking. $\neg \text{Texp} \leftarrow \text{SCT} \vee C1$ from identity rules[2], since C is not violated from the assumption of integrity before the transaction. A strengthen of our method is the more generated number of useful integrity tests than the previous methods like in [McC95]. The following sufficient tests are generated for the previous example.

$$\neg \text{Texp} \leftarrow \neg \text{Job}(J, \text{'Director'}) \vee S(C) \vee \text{Job}(J, \text{'Senior Secretary'})$$

$$\vee \text{Offering}(x, y, z) \wedge \text{Job}(y, \text{'Director'}) \wedge \neg S(x) \quad /*\text{assuming } C1 \text{ is not violated before the transaction}*/$$

$$\vee \text{Offering}(C, y, z) \wedge \text{Job}(y, \text{'Director'}) \vee \text{Offering}(x, J, z) \wedge \neg S(x) \vee \text{Offering}(C, J, z)$$

Since all the tests distributed over OR, the satisfaction of any disjunct alone is sufficient for integrity. $\neg \text{Job}(J, \text{'Director'}) \vee S(C) \vee \text{Job}(J, \text{'Senior Secretary'})$ is the complete and sufficient tests, $\text{Offering}(x, y, z) \wedge \text{Job}(y, \text{'Director'}) \wedge \neg S(x)$ is the original constraint, $\text{Offering}(C, y, z) \wedge \text{Job}(y, \text{'Director'})$ and $\text{Offering}(x, J, z) \wedge \neg S(x)$ are subsumed tests. Hence, $\text{Offering}(C, J, z)$ is the only new sufficient test.

5 Conclusion

Increasing the semantic content of the database model and a separate integrity maintenance subsystem are two approaches to maintaining integrity in database systems. The former leads to additional complexity for the users. The later creates additional overheads for the system. Separating integrity maintenance subsystem is more useful in minimizing the complexity faced by the users, since the overhead on the system can be managed and carefully optimized. It detects errors caused by database update operations and computes the repairs for these errors. The computed repairs are attached to the original erroneous update operation to create a correct and complete update operation. Our approach generates all minimal repairs to be presented to the user or the system administrator to select one of them to correct the update operation.

References

- [1] Ceri, S., Widom, J.: Deriving Production Rules for Constraint Maintenance. In: Very Large Data Bases Conference, vol. 16, pp. 566–577 (1990)
- [2] Chang, C., Lee, R.C.: Symbolic Logic and Mechanical Theorem Proving. Academic Press (1973)
- [3] Christiansen, H., Martinenghi, D.: On simplification of database integrity constraints. *Fundamenta Informaticae* 71(2), 371–417 (2006)
- [4] Ceri, S., Fraternali, P., Paraborch, S., Tanca, L.: Automatic Generation of Production Rules for Integrity Maintenance. *ACM Transaction Database Systems* 19(3), 366–421 (1994)
- [5] Gertz, M., Lipeck, U.W.: Deriving Integrity Maintenance Triggers From Transaction Graphs. In: Ninth IEEE Conference Data Eng., pp. 22–30 (1993)
- [6] Grefen, P.W.P.J.: Integrity Control in Parallel Database Systems. PhD Thesis, University of Twente (Netherlands) (October 1992)
- [7] Hanandeh, F., Ibrahim, H., Mamat, A., Johari, R.: Virtual rule partitioning method for maintaining database integrity. *Int. Arab J. of Information Technology* 1(1), 103–108 (2004)
- [8] Ibrahim, H.: Semantic Integrity Constraints Enforcement for Distributed Database. PhD Thesis, University of Wales College of Cardiff, Cardiff (UK) (June 1998)

- [9] Ibrahim, H.: A Strategy for Semantic Integrity Checking in Distributed Databases. In: Proceedings of the 9th International Conference on Parallel and Distributed Systems (ICPADS 2002), Taiwan, December 17-20, pp. 139–144 (2002)
- [10] Ibrahim, H.: Extending Transactions with Integrity Rules for Maintaining Database Integrity. In: Proceedings of the International Conference on Information and Knowledge Engineering, Las Vegas, USA, June 24-27, pp. 341–347 (2002)
- [11] Martinenghi, D.: Advanced techniques for efficient data integrity checking. PhD dissertation, Roskilde University, Denmark (2005)
- [12] McCarroll, N.F.: Semantic Integrity Enforcement in Parallel Database Machines. PhD Thesis, Department of Computer Science, University of Sheffield, Sheffield (UK) (May 1995)
- [13] Moerkotte, G., Lockemann, P.C.: Reactive Consistency Control in Deductive Databases. *ACM Trans. Database Systems* 16(4), 670–702 (1991)
- [14] Schewe, K.D., Thalheim, B., Schmidt, J.W., Wetzel, I.: Integrity Enforcement in Object Oriented Database. In: *Modeling Database Dynamics*, pp. 174–195 (1993)
- [15] Soumya, B., Madiraju, P., Ibrahim, H.: Constraint optimization for a system of relational databases. In: *Proc. of the IEEE Int. Conf. on Computer and Info. Technology*, Sydney, pp. 155–160 (2008)
- [16] Urban, S.D., Delcambre, L.M.: Constraint Analysis: A Design Process for Specifying Operations on Objects. *IEEE Trans. Knowledge and Database Eng.* 2(4), 391–400 (1990)
- [17] Urban, S.D., Lim, B.B.L.: An Intelligent Framework for Active Support of Database Semantics. *Int’l J. Expert Systems* 6(1), 1–37 (1993)
- [18] Wuethrich, B.: On Updates and Inconsistency Repairing in Knowledge Bases. In: *IEEE Conference of Data Eng.* (1993)
- [19] Wang, X.Y.: The Development of a Knowledge-Based Transaction Design Assistant. PhD Thesis, Department of Computing Mathematics, University of Wales College of Cardiff, Cardiff, UK (1992)

Optimization Algorithm for the Preservation of Sensor Coverage

Codruta-Mihaela Istin, Horia Ciocarlie, and Razvan Aciu

Department of Computer and Software Engineering,
“Politehnica” University of Timisoara
Blvd Vasile Parvan, Nr. 2, Postcode 300223,
Timisoara, Romania
istin.codruta@gmail.com,
horia@cs.upt.ro,
razvanaciu@yahoo.com

Abstract. Sensor networks that collect traffic information for supervision and immediate intervention are largely used. The domain of real time coverage algorithms is still an issue of debate. In order to achieve good coverage, the simulation phase is of major importance. Simulation can partially validate the performance of the algorithms and their dependability. This chapter presents a novel algorithm for coverage maintenance using driving prediction and the simulated results that validate its performance. Also, issues such as driving behavior have been considered.

1 Introduction

Several attempts regarding traffic models have been done, but there are still plenty of improvements that can be added. In general, for modeling highway traffic, Gaussian densities [1] are used. Another approach is the conditional autoregressive model. This model uses the Markov property [2]. In general, the concept of adjacently is considered to be represented by the situations from a certain segment defined as s . According to [3], the model assumes that the volume y observed at a location s obeys the formula:

$$y(s) = \epsilon_s + \sum_{r \in N(s)} \theta_r^s y(r) \quad (1)$$

where $N(s)$ represents the neighborhood of s , ϵ_s is considered additive noise and θ is a parameter calculated with ridge regression procedure [4]. The authors of [3] try to overcome the Gaussian densities drawback using Bayesian networks [5] that allow a certain degree of dependences. Their results show the improvement compared to the Gaussian initial model, but still, the proposed method addresses only to the car following models.

In this work we present a mathematical model for traffic prediction and coverage tested by Monte Carlo simulation. In this model we consider cars as events. An event can be formed by more than one car. If two or more cars are on the same

lane, have the same speed and the distances between them are less or equal to the minimum safe distance between two cars, those vehicles are considered part of the same event.

In our previous work [12,13] we presented an algorithm for coverage preservation in the presence of dynamic obstacles, in our case, vehicles. Sensors decided for themselves if they are obstructed in such a proportion that they became unuseful and turned their camera off. Before turning off, they searched in their redundancy group the most redundant camera to turn on in order to maintain a certain degree of coverage. The sensor that was initially turned off due to the obstruction was immediately turned on after the vehicle (dynamic obstacle) has passed.

The algorithm we propose in this work is an optimization of our previous work due to two important factors: sensors consider events as being the dynamic obstacles and sensors stay off all the predicted obstruction period. These facts have as a result an optimization in power consumption. The sensors that turn off due to obstruction will turn on again only after the whole event has passed. Furthermore a comparison between the algorithm presented in this chapter and the algorithms from previous work is shown.

The authors of [6] propose a car following model based also on the fluid dynamics accordingly defining a relation between speed and density. This model is called the car following model and was first introduced in [7]. The model was improved by adding the safe distance concept that directly influences the velocity of the vehicles was introduced in [8]. The mesoscopic models compare the interactions between particles of a gas to the interactions of vehicles on a road. Based on this idea the authors of [9] mathematically model the concept of acceleration and overtaking behaviors and obtain the critical density of the phase transition from free flow congestion. References [10, 11, 15] propose models for estimating the travel traffic delays also considering the congestion probability. Based on these studies, the effects of introducing traffic lights in different intersections were analyzed from the decongestion and waiting time perspective.

2 Problem Description

Coverage preservation realized by a good sensor management is tight to their response to the traffic flow that represents the dynamic obstacles. As shown in chapter 2, there are several approaches to solve this problem. It still remains an issue especially if the purpose of the model is maintaining a certain degree of coverage. According to [14] there is still a gap between microprogramming and macro programming a Wireless Sensor Network (WSN). We try to overcome this gap by introducing the concept of event combined with traffic prediction. The need of introducing this concept came exactly from the need to create a bridge between micro and macro level in a WSN. The micro perspective means that the behavior of each car is observed. The other approach is to have a global view from the beginning.

The algorithm presented in our previous work [12] is optimized. There were some drawbacks that were resolved by the use of introducing new concepts and also by combining the micro and the macro traffic. A better performance is

obtained if a sensor stays off until the whole event passes, even though an event can be formed by several vehicles. It is obvious that turning the sensor on and immediately turning it off again would be a waste of energy with very little gain. The gain would be the visibility of the area between cars, but that area might already be covered by another sensor. Furthermore, the speed of the event can be high, so the area that might have been seen if the sensor would have been turned on would have been a short glimpse. Gathering the data in such a short time and process it, it is also difficult to accomplish. For these reasons we considered that this approach is an improvement.

Another significant advantage is the prediction aspect. Sensors still turn off when they are obstructed and they become unuseful, but they are now able to compute when the event would have passed and turn themselves on again. Proceeding this way, more energy is saved without significant loss in coverage.

The model we propose in order to have a good prediction is a mix between the micro and macro perspective. The idea behind this is that when sensors are turned on, they see cars and they register their speed, lane and orientation. We will explain how this is realized later in the work. The characteristics of cars mentioned above are registered in a global database together with their offsets. If the conditions are fulfilled, events are formed. If a sensor is turned off due to obstruction, it looks in the database to see the characteristics of the event and it turns itself off on the period it predicts the event will by still obstructing for itself. After the predicted period passes, it will automatically turn itself on, again.

This method has both a distributed and a centralized component. The distributed component is the part in which the sensors decide if they are obstructed and become unuseful so they turn themselves off, the sensors compute the duration of the obstruction, and the sensors turn themselves on again after the event has passed. The centralized component is necessary because all the information that sensors compute, are registered on a server and all the sensors have access to that global database.

Sensor Capabilities – The method proposed has only been tested by different types of simulation (see Section 4), but the target is to implement and test this algorithm in practice. In order to do this the sensors must have some capabilities: they must have a video camera that has an optical focus capacity, an internal clock and wireless transmission capacity.

Initially, after deployment, on the lanes of interest a special car will be driven that will help sensors' calibration. This car will have a known constant speed that will emit the code corresponding to the lane it drives on. This car will also have a pattern drawn on it (for example two big spots - one color in front and another color at the back) in order to distinguish the direction of the lane. Furthermore, computing the data offered by the calibration car such as the lane, its direction, and also considering the velocity of the car, relationships regarding the relative positions between sensors can be determined.

The simplest methods that sensors could use in order to detect the calibration car and to recognize its pattern are phase detection and contrast measurement. Phase detection is similar with the way eyes form their image. Two points are

needed and the image is then formed. Contrast measurement is based on computing the blur of the image. When the contrast between pixels is maximum, the blur is minimum, so the image has the best quality. This method is easy to realize and the sensor can compute its blur when the calibration car drives in front of it. After the sensors have been calibrated, they will be able to distinguish the lane and event differences.

3 Algorithm Description

Event concept – Vehicles that drive on the same lane and have the distance between them less or equal to the minimum allowed distance between two vehicles form an event. The event is not restricted by its length. An event can be formed by coupling two or more events or can be split in the conditions mentioned below. A single vehicle is also seen as an event.

Model Description – When a sensor is on, it registers in a database all the cars that appear in its visual area. More specific, each sensor sends to the database the lane, the speed of the event and the starting and ending points of the segments defining the car. The server then computes the properties of each vehicle and if the conditions are fulfilled, from separate segments defining particular cars, events are formed. If a car is driving at a certain distance from the other cars, that car will form an event by itself and the car that the sensor had registered will remain a singular event until the conditions will be accomplished and will be coupled with another event.

If two events are on the same lane and the distance between the events is less than d_{min} (safe distance between cars), the events will be concatenated and the result will be one single concatenated event. The extremities of the composed event are given by the minimum and the maximum values on the OX axes from all the unique segments that are contained in the composed event.

We note f (front) the margin of a segment that has the greatest offset on the lane and we note b (back) the margin of the same segment that has the minimum offset on lane. We consider that the offset is computed on the direction of event propagation. In this case we have:

$$f_E = \max(f_i), \quad f_i \in F_E, \quad b_E = \min(b_i), \quad b_i \in B_E$$

Where f_E is the front of the event, b_E is the back of the event, F_E is the set of all front segments frontiers, B_E is the set of all back segments frontiers.

Considering these notations, two events E_1 and E_2 will be concatenated in one of the following situations:

- $f_{E_1} < b_{E_2}, d(f_{E_1}, b_{E_2}) \leq d_{min}, v_{E_1} < v_{E_2} \Rightarrow E\{b_{E_1}, f_{E_2}, v_{E_2}\}$
- $f_{E_2} < b_{E_1}, d(f_{E_2}, b_{E_1}) \leq d_{min}, v_{E_2} \geq v_{E_1} \Rightarrow E\{b_{E_2}, f_{E_1}, v_{E_1}\}$

There is one more concatenation situation when a car is on a separate lane and it changes the lane, coming into a column of cars that form an event. In this situation the vehicle is just inserted into the event. Nothing else changes due to the fact that the extremities of the event remain unchanged. The speed v also remains the same

because if the vehicle that was inserted into the event had to have the same speed as the event in order to be integrated in it, where E is the new event resulted by the concatenation of the two, $d(a,b)$ is the distance from point a to point b, d_{min} is the minimum distance between cars.

The events are concatenated if they are on the same lane and are kept in a common database that is available to all sensors (s). The events registered in the common database are constantly normalized to the current time t_{crit} by the propagation of the event in time. The propagation is computed on a certain segment with respect to the speed v of the event and to the difference in time from the last propagation of the same event. We note t_{DB} the current time of the event in the database, t_C the current time, v_{DB} the speed of the event in the database and o_{DB} the offset of a segment that is part of the event from a lase. This offset is useful at t_{DB}

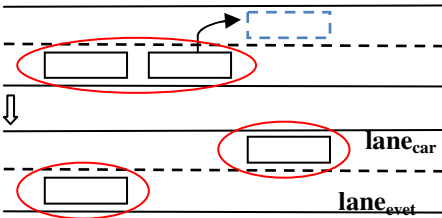
By normalization we will update the offsets of all segments and in the same time the time of the last propagation of the event in the database in order to simulate the movement of the event in the database according to its speed v . The result will be:

$$t'_{DB} = t_C, \quad o'_{DB_i} = o_{DB_i} + (t_C - t_{DB}) * v_{DB}, \forall o_{DB_i} \in o_E,$$

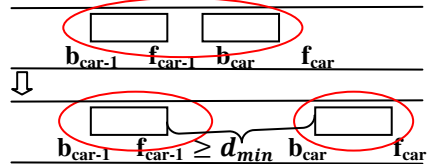
○ o_E is the set of all segments' offsets from the event, t'_{DB} is the new current time from the database, o'_{DB_i} is the new offset of the segment from the event.

Once an event is registered in the database, for a simulation step, its lane or speed will not be modified. If a vehicle that was previously registered in an event is noticed outside an event, the situation is updated in the database with respect to the new real situation. From this update a modification of the segments' position in the event can appear or it is also possible that the segment is completely removed from the event, if the conditions are not fulfilled anymore. In this case a separation of a vehicle from an event can take place in one of the following situations:

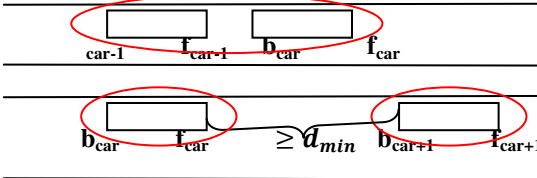
1. $lane_{car} \neq lane_E$



2. $f_{car} = f_E$
 $v_{car} > v_E$
 $d(b_{car}, f_{car-1}) \geq d_{min}$

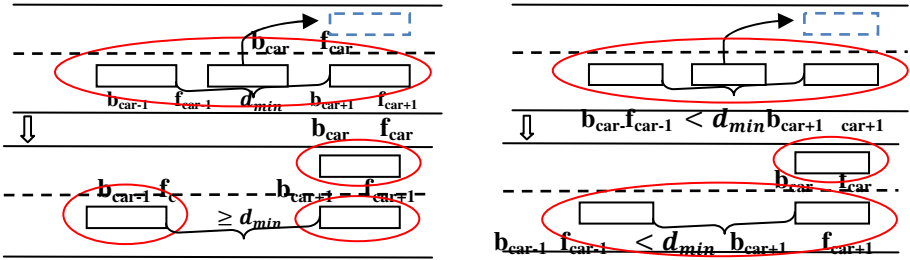


3. $b_{car} = b_E$
 $v_{car} < v_E$
 $d(f_{car}, b_{car+1}) \geq d_{min}$ where



- b_{car+1} and b_{car-1} have been noted the cars that are in front and after the considered car

We can distinguish the next possible situations:



- $d(f_{car-1}, b_{car+1}) < d_{min}$
the event remains the same with the exception that the segment corresponding to the considered car is removed from the event
- Corresponding to 1. if $d(f_{car-1}, b_{car+1}) > d_{min}$
 - ⇒ $E_1\{b_E, f_{car-1}, v_E\}$ and $E_2\{b_{car+1}, f_E, v_E\}$
- b) Corresponding to 2.
 - ⇒ $E\{b_E, f_{car-1}, v_E\}$
- c) Corresponding to 3.
 - ⇒ $E\{b_{car+1}, f_E, v_E\}$ and $E'\{b_{car}, f_{car}, v_{car}\}$

To this event, a new event will be added corresponding to the considered car. If a turned on sensor realizes that it is obstructed in a proportion greater than the maximum given obstruction, it is considered that the sensor becomes useless and it will turn off.

For all the turned on sensors, a coverage analysis is performed and if two sensors that are turned on and those sensors are in the same redundancy group, the sensor that covers the less, will be turned off. The sensors that are off will be turned on again if two conditions are fulfilled: there is no sensor that has a better coverage than itself that is on and is part of its redundancy group and the fact that it is not obstructed in a greater proportion greater than the limit obstruction level by any events registered in the common database

Due to the fact that the sensors that are turned off, they cannot compute their real obstruction with respect to the vehicles that are in front of it. In this case, the obstruction will be computed taking into consideration the speed, the position and the length of the event. A composed event is considered to be obstructing on all its length on the road because the spaces between the segments are too small at those speeds and the processing capacity of the sensors is too small in order to be able to register possible events that are not on the closest lane to the sensor.

An important concept that is used in managing sensors in the current algorithm is redundancy. As we mentioned earlier, two sensors that are redundant cannot be on in the same time. A sensor is considered to be redundant with another sensor if they cover the same area in a proportion greater than an established limit. This

concept is more detailed in our previous work [12]. In the current work we defined the model that describes traffic also from a global view. This way a more accurate simulation based on events and on the idea of event propagation is realized.

4 Test Results

The experiment section shows the efficiency of this algorithm in comparison with other developed algorithms. The current method was implemented as a Java program and run on a desktop PC. This section presents a comparison between the described algorithms and the algorithms from [13] where first are no algorithms were applied to the WSN and where the algorithm where the traffic is analyzed at micro level. This means that no composed events are considered. Data sets used for testing are real.

The traffic cases were generated with a generator also implemented as a Java program. Traffic was generated according to real situations. The method used in simulating traffic is called Monte Carlo simulation. A more detailed description of the Monte Carlo simulation that was used can be found in [13]. Still it is important to emphasize the fact that each Monte Carlo simulation lasted for 1 day in order to analyze all traffic situations [17,18]. The graphics presented below are snapshots of the worst situations found. In addition to this, we simulated how the current algorithm would work if we considered that sensors have limited battery. We did this by allowing the sensors a limited time in which each sensor could be turned on and perform traffic surveillance.

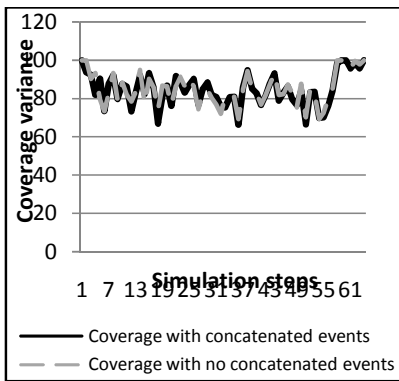


Fig. 1 Comparison between coverage levels with concatenated and no concatenated events

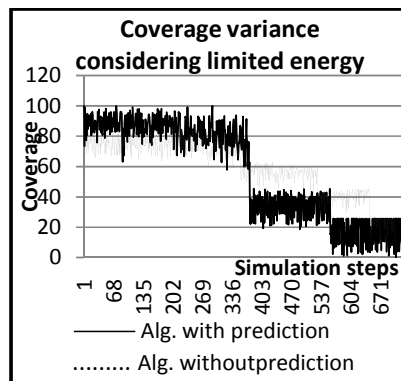


Fig. 2 Coverage variance considering limited energy

It can be observed in Figure 1 that the difference between the case when the cars are concatenated forming composed events and when the cars are not concatenated and each car is a single event is really small. As expected, when events are not composed, the coverage is better due to the fact that sensors turn on and off

after each single event, meaning after each vehicle. It is important to notice that the minimum coverage value in the concatenated events case is 66.303% and in the case of single events is 69.363%.

The prediction algorithm always performed better than the algorithm without prediction. In Figure 2 can be observed that in the first part, the prediction algorithm performs better, but in the second half, the algorithm using redundancy groups performs better. The explanation is the fact that in the first part, for both algorithms, sensors have energy and the performance of the prediction algorithm is better. As presented at the algorithms description at both algorithms if they have energy resources, the best sensors are used, so the algorithm performance is their best. After energy starts to finish those sensors are replaced by other sensors that do not perform as well as did the ones that consumed their battery. The performance of the algorithm without prediction is better in the second part due to the fact that the algorithm tries to turn on sensors from the redundancy group of the sensor that drained its battery, so the algorithm tries to turn on sensors from each redundancy group and as a result the algorithm offers as good coverage as possible along the entire road.

Figure 3 shows the comparison between the performance of the current algorithm and the algorithm presented in our previous work [12,13]. It was described the fact that not all sensors are turned on and the reason was shown. In the previous algorithm, the maximum number of sensors that were on at a time with no redundancy between them was computed at the level of redundancy groups. The idea was that the sensors communicated between them in their redundancy groups in order to determine which sensor to turn on. Due to the fact that in a redundancy group, the distances between the sensors are relatively small, the energy spent on communication was not considered to be an issue. Furthermore, due to the fact that not all the sensors are on, the best coverage situation given by the maximum number of sensors on from each redundancy group is considered to be the optimal situation and was calibrated in our previous tests at 100%. The difference shown in Figure 2 comes from the fact that the calibration was computed with respect to the highest coverage value from both sets of data: the old algorithm and the new one.

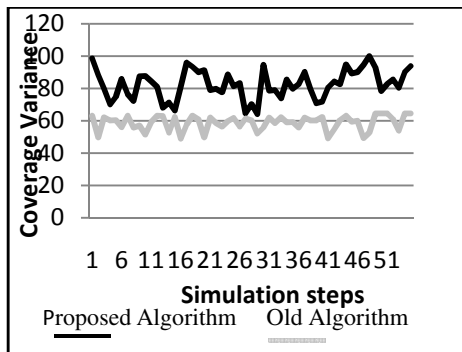


Fig. 3 Comparison between the proposed algorithm and the algorithm presented in [12]

It is obvious that the present algorithm performs much better than the older one. This comes from the fact that in the current algorithm, the maximum number of sensors that have no redundancy between them at the level of the network is less. This means that in the current algorithm, the number of sensors that are on at a time is higher and the coverage obtained is better.

Both algorithms have to constantly communicate with the central unit in order to provide real time data and also to request information about the status of the whole network, more in the current algorithm regarding the event prediction and in proportion also regarding the sensor management.

Figure 4 and Figure 5 presents the influences of the driving manner upon coverage determined by both algorithms. In traffic drivers have behave differently. In our work we divided the behavior into normal behavior and speedy behavior. Studies of reckless driving behavior have been made in [16]. The simulator acts different if the vehicles' behavior is speedy. The differences are first of all the speed that is significantly higher than the normal average speed. Also besides the speed, if the simulator determines that a vehicle is speedy, it allows that vehicle to overtake on the right side, if a minimum safe distance between cars allows it. Moreover, for speedy vehicles, the simulator also adjusts the minimum distance between cars by reducing it with 20% from the normal distance between cars applied at normal behavior.

The results show a maximum variation of about 2% for both algorithms. There is no noticeable difference between the performances of the algorithms.

The variation is more dependent of the parentage of speedy cars than of the minimum distance between cars. The cause is related to the acquisition and processing time of the sensors, so as the car speed increases, the sensor analysis quality decreases. When tested on different traffic behavior situations, both algorithms (with and without prediction) show very good stability. The test cases included variations of percentage of speedy cars, between 20–60% and of the minimum distance between cars 7-13 m. For testing we used the case with limited amount of energy for the sensors. The metric used was total coverage (TC): $TC = \int_{t=t_0}^{t_f} C(t)$ lwhere TC – total coverage, C(t) – coverage at moment t, t0 – simulation starting time, tf – simulation ending time, when no energy is left at any sensor.

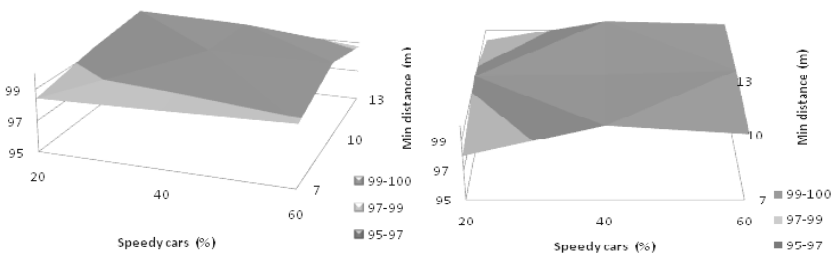


Fig. 4 Coverage variance for algorithm without prediction

Fig. 5 Coverage variation with prediction

5 Conclusion

The presented algorithms were developed as solutions to traffic surveillance using wireless sensors. Both solutions work and are perform well. Between the two algorithms there is a tradeoff between coverage and energy consumption. The level of energy consumption was not simulated but the remark comes from the fact that more requests to the central unit mean more energy consumption. We intend to test the two algorithms with respect to their energy consumption to see how big the tradeoff between coverage and energy is, but the object of the current work was to show the coverage performance of the new algorithm and we can conclude that the performance is high.

References

- [1] Belomestny, D., Jentsch, V., Schreckenberg, M.: Completion and continuation of non-linear traffic time series: a probabilistic approach. *Journal of Physics* 36, 11369–11383 (2003)
- [2] Lauritzen, S.L.: *Graphical Models*. Oxford University Press (1996)
- [3] Singliar, T., Hauskrecht, M.: Modeling Highway Traffic Volumes. In: *Proceedings of the 18th European Conference on Machine Learning* (2007)
- [4] Hastie, T., Tibshirani, R., Friedman, J.: *Elements of Statistical Learning*. Springer (2001)
- [5] Jensen, F.V.: *An Introduction to Bayesian Networks*. Springer, New York (1996)
- [6] Baykal-Gursoy, Duan, Z., Xu, H.: Stochastic Models of Traffic Flow Interrupted by Incidents. *Control in Transportation Systems* (2010)
- [7] Richards, P.I.: Shock waves on the highway. *Operations Research* 4, 42–51 (1956)
- [8] Helbing, D., Hennecke, A., Treiber, M.: Phase diagram of traffic states in the presence of inhomogeneities. *Phys. Rev. Lett.* 82, 4360 (1999)
- [9] Prigogine, I., Andrews, F.C.: Boltzman-like approach for traffic flow. *Operations Research*, 789–797 (1960)
- [10] Helbing, D.: A section-based queuing-theoretical traffic model for congestion and travel time analysis in Networks. *Journal of Physics* 36, L593 (2003)
- [11] Lammer, S., Donner, R., Helbing, D.: Anticipative control of switched queueing systems. *Euro. Physics. J. B* 63, 341–347 (2008)
- [12] Istin, C., Pescaru, D., Ciocarlie, H., Curiac, D., Doboli, A.: Reliable Field of View Coverage in Video-Camera based Wireless Networks for Traffic Management Applications. In: *IEEE Symposium on Signal Processing and Information Technology, ISSPIT* (2008)
- [13] Istin, C., Pescaru, D., Ciocarlie, H., Doboli, A.: Monte-Carlo Simulation of a Dynamic Coverage Preservation Algorithm for Video Sensor Networks. In: *Fifth International Conference on Dependability of Computer Systems, DepCoS-RELCOMEX* (2010)
- [14] Gaura, E., Brusey, J., Halloran, J., Daniel, T.: Distributed Information Extraction from Large-scale WSNs. In: *Approaches and Open Research Issues, Sensors & Transducers, IFSA* (2012) ISSN 1726-5479

- [15] Ammar, K., Nascimento, M.A.: Histogram and other aggregate queries in wireless sensor networks. In: Bayard Cushing, J., French, J., Bowers, S. (eds.) SSDBM 2011. LNCS, vol. 6809, pp. 527–536. Springer, Heidelberg (2011)
- [16] Booyesen, M.J.: Regional ITS Perspectives - The African Market. In: Fully Networked Car Conference, Geneva (2012), <http://goo.gl/GzUbs>
- [17] Stanley, J.: Passenger Car Equivalents of Trucks Under Lane Restriction and Differential Speed Limit Policies on Four Lane Freeways. PhD thesis (2009)
- [18] Croney, P., Croney, D.: The design and performance of road pavements, 3rd edn. McGraw Hill (1997)

Methods for Detecting and Analyzing Hidden FAT32 Volumes Created with the Use of Cryptographic Tools

Ireneusz Józwiak, Michał Kędziora, and Aleksandra Melińska

Wroclaw University of Technology, Wyspiańskiego 27 50-370 Wrocław, Poland
{Ireneusz.jozwiak,michal.kedziora,
aleksandra.melinska}@pwr.wroc.pl

Abstract. The article describes the theoretical and practical methods for detecting and analyzing hidden volumes created with the use of cryptographic tools. The presented method is based on an analysis of the differences that result from the use of a hidden volume in FAT32 file systems. The method is effective both when the password is known to the host container and in the situations when password is not known. Potential computer forensic application of this methodology varies from standard investigations to advanced analysis of network and in the cloud data storages.

1 Introduction

One of the most common technique used to defeat computer forensic investigation process is using data encryption[1]. Using encryption has sufficient effect in forensic investigations and it is formally named as one of anti-forensic techniques [2]. Furthermore it becomes security standard in the network applications and cloud storage. No plaintext essential data should be sent to network due to possibility of interception using network sniffers. Also to achieve higher level of security in a cloud storage, data can be encrypted before placing it into the cloud. Also it is good security practice and it is recommended to encrypt all storage disks to achieve security in the case of physical access to device. It is recommendation not only for laptops but also for every storage. On the one hand using encryption is leading to higher data security against information leakage, but on the other hand causes series of issues during computer forensic investigations [3].

Standard cryptography is supposed to keep data secure for parties which possess encryption key (e.g. password) [9]. In fact computer forensic investigators has several options to easily and successfully recover a encryption key [4]. The encryption can be found in plaintext in the computer documents, configuration files or paper documents. Passwords most often can be easily brute forced if users do not follow security policies. Research indicates that most users use the same password to protect more than one asset [16]. This mean that it can happen that

e.g. password from mail box or web page which is easy to recover, will be the same as password to encrypted container. The practice shows that, the password is often revealed by user when law enforcement officer asks. Furthermore in some countries (e.g. United Kingdom) not revealing password to authorities is treated as a crime and can lead to a sentence of 5 years in prison for failing to comply with police. In the United Kingdom 5 years imprisonment is reserved for terrorism cases, and 2 years sentence in any other cases [8]. All this issues involving potential danger of revealing password led to developing deniable cryptography.

A deniable cryptography is method described by Rafail Ostrovsky in the paper called "Deniable Encryption" [12]. Deniable encryption allows cipher text message to be decrypted to different plaintexts, using different encryption keys. This allows user to hide his secret message even if he is forced to reveal the password. In the basic case for pre-shared key encryption a protocol π with sender S and receiver R and with security parameter n , is a shared key $\delta(n)$ deniable encryption protocol if the probability that R output is different than S is negligible. There is an algorithm ϕ having property that for any, let k, r_S, r_R be uniformly chosen shared key and random input S and R. Also let $c = \text{com}_{\pi}(m_1, k, r_S, r_R)$, and let $(k', r'_S) = \phi(m_1, k, r_S, c, m_2)$. Then the random variables are:

$$(m_2, k', r'_S, C) \text{ and } (m_2, k, r_S, \text{com}_{\pi}(m_1, k, r_S, r_R)) \quad (1)$$

Resuming, when dealing with cryptography in the computer forensics applications, most demanding issue is not to find encryption key, but to detect potential use of deniable encryption.

2 Threat Model

Bruce Schneier in paper "Defeating Encrypted and Deniable File Systems: TrueCrypt v5.1a and the Case of the Tattling OS and Applications" defined three potential threat models against which encrypted container with hidden volume should be secure [11].

First model is named as "one-time access". One time access is a model in which attacker has only one binary copy of the disk image containing encrypted container or just a container themselves. This situation is common in computer forensic investigations when hard drive is seized by forensic experts and then investigated. This is also most demanding and hard situation when trying to detect hidden volume. There are ways to detect hidden volumes when dealing with this model using artifact analysis, hash analysis or metadata analysis. But in our work we will focus on other model that includes intermittent and regular access to suspect's data.

Intermittent and regular access is a model that describes the situation when the attacker has several copies of encrypted container, taken at different times. This situation is best described by this model is seizing backups which contain encrypted container snapshot from different dates. Two research problems we have to deal in this model detect potential encrypted containers and detect possible use of Deniable File System inside encrypted container.

3 Entropy Based Container Detection

Most of the encryption detection algorithms are depended on calculating entropy. Originally entropy definition comes from thermodynamics and is a measure of the disorder or randomness of the constituents of a thermodynamic system. Entropy was adopted into computer science and represents measure of the uncertainty associated with a random variable [10]. Entropy H of a discrete random variable X with possible values $\{x_1, \dots, x_n\}$ is equal:

$$H(X) = E(I(X)) \tag{2}$$

Where I is the information content of X. $I(X)$ is a random variable and E is the expected value. If p denotes the probability mass function of X then entropy is equal to:

$$H(X) = \sum_{i=1}^n p(x_i) I(x_i) = - \sum_{i=1}^n p(x_i) \log_b p(x_i), \tag{3}$$

where b is the base of the logarithm. Entropy value will be close to max value when the input will be random data. Any signs of data order will lower entropy value. In practical applications forensic tools use Chi square based detection algorithm which is much more accurate [15].

Chi Square test is a statistical hypothesis test in that the distribution of the test is a chi-square distribution and the null hypothesis is true. There are several tests build on that assumption, but main use is to confirm randomness of data. From the definition chi-square test with k probable outcomes, performed n times, in which $Y_1, Y_2, Y_3, \dots, Y_k$ is the number of experiments which resulted in outcome, where the probabilities of each outcome are p_1, p_2, \dots, p_k is:

$$x^2 = \sum_{1 \leq s \leq k} \frac{(Y_s - np_s)^2}{np_s} \tag{4}$$

We should expect the lower chi square sum for more random data [14]. From a chi-square, the probability Q that the X^2 sum for the test with d degrees of freedom is regular with null hypothesis and can be compute as:

$$Q_{x^2, d} = \left[2^{d/2} \Gamma\left(\frac{d}{2}\right) \right]^{-1} \int_{x^2}^{\infty} (t)^{\frac{d}{2}-1} e^{-\frac{t}{2}} dt \tag{5}$$

Where Γ is a factorial function to complex and real arguments:

$$\Gamma_x = \int_0^{\infty} t^{x-1} e^{-t} dt \tag{6}$$

This algorithm can be used by computer forensic investigators to point encrypted containers. Next step is to detect is there any hidden volume in use inside encrypted container.

4 Hidden Volume Detection and Analysis

Standard block ciphers are using two inputs. First input used in block ciphers is a key $K \in \{0,1\}^k$ and a message $M \in \{0,1\}^n$ and produce output ciphertext $C \in \{0,1\}^n$, Description of block cipher can be written as:

$$E : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n. \quad (7)$$

Since 2010 major cryptographic tools are using XTS (XEX-based tweaked-codebook mode with ciphertext stealing) to create encrypted containers (e.g. DiskCryptor, BestCrypt, dm-crypt, FreeOTFE, TrueCrypt...). XEX-based tweaked-codebook mode with ciphertext stealing is encryption type based on tweakable encryption mode XEX invented by Phillip Rogaway[5]. Tweakable block cipher can be described formally as:

$$\tilde{E} : \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \rightarrow \{0,1\}^n. \quad (8)$$

Tweakable block cipher takes three inputs: besides a Key $K \in \{0,1\}^k$ and Message $M \in \{0,1\}^n$, also takes a tweak $T \in \{0,1\}^t$ to produce cipher text $C \in \{0,1\}^n$. It is shown on figure 1(b). The tweak T in XEX-based tweaked-codebook mode with ciphertext stealing is represented as a combination of the sector address and index of the block inside the sector [7]. This implementation limits the maximum size of each encrypted data units to 2^{20} blocks (according to XTS-AES standard).

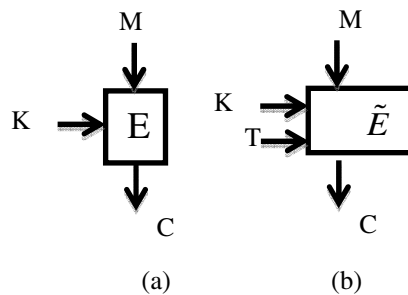


Fig. 1 (a) block cipher encrypts message M with key K into ciphertext C. (b) Tweakable block cipher encrypts message M with key K and tweak K into Ciphertext C.

Method” to create hidden volume inside encrypted containers. We decided to this solution because this method of creating hidden volumes is we believe most common used in practice. After compering original encrypted containers without hidden volume with those containing hidden volume we discovered differences in certain part of containers. Example of that comparison is presented on Figure 2. Different binary values are marked darker color. We analyze occurrence of blocks with different values and place it into table 1. In first column are marked different characteristic points and areas of encrypted containers.

Table 1 Characteristic points of hidden volume creation process

Container area	F_s	F_m	F_b
X_{diff1} start	10000	10000	10000
X_{diff1} end	101FF	101FF	101FF
X_{diff1} length	$(511)_{10}$	$(511)_{10}$	$(511)_{10}$
X_{diff2} start	44000	5FFFD	E0000
X_{diff2} end	487FF	64FFF	E5BFF
X_{diff2} length	$(18431)_{10}$	$(20482)_{10}$	$(23551)_{10}$
X_{diff3} start	6D000	F0000	1F0000
X_{diff3} end	6D1FF	F01FF	1F01FF
X_{diff3} length	$(511)_{10}$	$(511)_{10}$	$(511)_{10}$
X_{EOF}	7CFFF	FFFFF	1FFFFF
$X_{EOF} - (X_{diff3} \text{ end})$	$(65024)_{10}$	$(65024)_{10}$	$(65024)_{10}$

In columns two to four are lengths and values for each of encrypted container. As we can see, despite of different sizes of encrypted containers we can observe some regularity. First observation is that independently from container size, when hidden volume is created an area with different values is created with the same starting point (offset 10000 equals 65536), same length of 511 Bytes. Second area with different values was observed but it had different size and starting points. Third area has length of 511 Bytes and ends 65024 bytes before the end of the file. Reassuming process of creating hidden volume using direct method in True Crypt can be easily detected by observing changes in area between X_{diff1} start and $X_{EOF} - (X_{diff3} \text{ end})$ according to:

$$(X_{d1s} \dots X_{d1e}) \text{ and } (X_{EOF} - (X_{diff3}(\text{end})) - X_{d31} \dots X_{EOF} - (X_{diff3}(\text{end})))$$

$$(10000 \dots 101FF) \text{ and } (FC01 \dots FE00)$$

Performed experiments confirmed that the only process of creating direct hidden volume creates that change to encrypted container. We performed experiments to confirm that any other operation on encrypted container produces changes in marked areas. The visualization of this we presented on fig. 3.

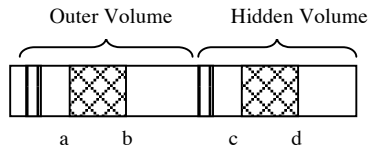


Fig. 3 Visualization of differential analysis of the container with both outer and hidden volume content change

For FAT32 file system copying files both to the outer volume or to the hidden volume creates similar footprint in encrypted container which consist of three small different areas on the beginning of the volume, and one big area which has size of file copied. On the figure 3 we present visualization of the encrypted container what is build from outer volume and potential hidden volume. During the testing we have copied file into both outer (b) and hidden volume (d). It turned out that thanks to FAT32 file system characteristics we can easily detect that file was copied into outer volume or hidden volume. Creating file in FAT32 file system creates new entry into File Allocation Table, this process after encryption looks like three small areas appeared on the start of the volume. This information is sufficient to determine existence of hidden volume from forensic point of view.

5 Summary

In the paper we presented methods of analyzing and detecting hidden FAT32 volume inside encrypted containers. After performing a series of tests it was succeed to point four areas of encrypted containers that are changed during process of creating hidden volume, and characteristic areas which are changing while hidden volume is used. Furthermore the areas appear at the same places despite various length of tested containers. The presented methods can be used in practice by forensic investigators to detect hidden volumes. This may increase quality and quantity of evidence found. The method described in the work is effective both when the password to the outer volume is known to the forensic investigator and in situations where password is not known. The further research will focus on analyzing hidden file systems with different file systems (especially NTFS), and we will focus for applying solution into the cloud storage forensic analysis.

References

- [1] Kornblum, J.D.: Implementing BitLocker Drive Encryption for forensic analysis. *Digital Investigation* 5(3-4), 75–84 (2009) ISSN 1742-2876
- [2] Casey, E.: Practical Approaches to Recovering Encrypted Digital Evidence. *International Journal of Digital Evidence* 1(3) (Fall 2002)
- [3] Altheide, C., Merloni, C., Zanero, S.: A methodology for the repeatable forensic analysis of encrypted drives. In: *Proceedings of the 1st European Workshop on System Security, EUROSEC 2008* (2008)

- [4] Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys (February 2008) (submitted for publication)
- [5] Liskov, M., Rivest, R., Wagner, D.: Tweakable Block Ciphers. MIT and UC Berkeley (2002)
- [6] Security in Storage Working Group of the IEEE Computer Society Committee. IEEE P1619, Standard for Cryptographic Protection of Data on Block-Oriented Storage Devices (2007)
- [7] National Institute of Standards and Technology (NIST). NIST Special Publication 800-38E, Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices (January 2010)
- [8] Information Technology Laboratory, NIST. FIPS Pub 140-2: Security Requirements For Cryptographic Modules
- [9] Kleidermacher, D.: Enhance system security with better data-at-rest encryption. Embedded (March 2012)
- [10] Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996) ISBN 0-8493-8523-7
- [11] Czeskis, A., St. Hilaire, D.J., Koscher, K., Gribble, S.D., Kohno, T., Schneier, B.: Defeating Encrypted and Deniable File Systems: TrueCrypt v5.1a and the Case of the Tattling OS and Applications. In: 3rd Workshop on Hot Topics in Security (2008)
- [12] Canetti, R., Dwork, C., Naor, M., Ostrovsky, R.: Deniable Encryption, Cryptology ePrint Archive, Report 1996/002
- [13] Anderson, R., Needham, R., Shamir, A.: The steganographic file system. In: Aucsmith, D. (ed.) IH 1998. LNCS, vol. 1525, pp. 73–82. Springer, Heidelberg (1998)
- [14] Jozwiak, I., Kedziora, M., Melinska, A.: Theoretical and practical aspects of encrypted containers detection - digital forensics approach. In: Zamojski, W., Kacprzyk, J., Mazurkiewicz, J., Sugier, J., Walkowiak, T. (eds.) Dependable Computer Systems. AISC, vol. 97, pp. 75–85. Springer, Heidelberg (2011)
- [15] Walker, J.: Introduction to Probability and Statistics, A Pseudorandom Number Sequence Test Program. Fourmilab (January 28, 2008)
- [16] Fragkos, G., Tryfonas, T.: A cognitive Model for the Forensic Recovery of End User Passwords. Digital Forensics and Incident Analysis, 48–54 (2007)

Critical Infrastructures Safety Assessment Combining Fuzzy Models and Bayesian Belief Network under Uncertainties

Vyacheslav Kharchenko¹, Eugene Brezhniev¹,
Vladimir Sklyar², and Artem Boyarchuk³

¹ National Aerospace University KhAI, Kharkiv, Ukraine,
Centre for Safety Infrastructure Oriented Research and Analysis
V.Kharchenko@khai.edu,
milestone@list.ru

² Research and Production Company Radiy,
Kirovograd, Ukraine
vvslyar@mail.ru

³ National Aerospace University KhAI, Kharkiv, Ukraine
a.boyarchuk@khai.edu

Abstract. The complexity of critical infrastructures (CI) and systems safety assessment calls for the need for integration of different methods that use input data of different qualimetric nature (deterministic, stochastic, linguistic). Application of one specified group of risk methods might lead to loss and/or disregard of a part of safety-related information. Bayesian Belief Network (BBN) and fuzzy logic (FL) represent a basis for development of the hybrid approach to capture all information required for safety assessment of complex dynamic system under uncertainties. Integration of FL-based methods and BBNs allows decreasing the amount of input information (measurements) required for safety assessment when these methods are used independently outside from the proposed integration framework. The processes of CI parameters' measurement might technically difficult and expensive. Instrumentation layer's operation might be compromised in emergency situations due to its dependence on power supply. The hybrid methods might be considered as basis for the expert system to help the operator make the decisions. The application of hybrid methods makes operator less dependent on information from instrumentation and control system (I&C). The illustrative example for Nuclear Power Plant (NPP) reactor safety assessment is considered in this chapter.

1 Introduction

The problem of CI and systems (as an example, NPP I&C systems) safety assessment is topical due to importance of current tasks. Thus, for example, CI safe operation is critical for the strategy of country industrial development and the

growth of welfare of its citizens. Fukushima-1 NPP accident showed that the CI reliability and safety level contributes to the public confidence in them, which in turn has a direct impact on the length of their life cycle, their modernization and reconstruction projects financing level.

The set of input data used in CI safety analysis includes: *deterministic data* (D_d). An information set giving a credible description of CI (specifications, operating parameters and modes, systems structure, etc.); *statistical (historical) data* (D_s), accumulated by observation of CI systems parameters throughout their life cycle. Into this category fit reliability and safety characteristics, operating environment conditions, external systems; *linguistic data* (D_L), *represented as natural language expressions*, obtained from professional experts in this field. A part of information about CI behavior may be represented in the form of expert knowledge, which should also be taken into account in CI safety assessment.

The problem of CI safety assessment can't be solved within the scope of one disciplinary approach. Consequently, in order to obtain a reliable safety values it is reasonable to use all the above groups of input data.

Analysis of literature shows lack of attention given to the issues of development of approaches to integration of different safety assessment methods. Thus, the work [1,2] suggests an idea to combine qualitative and quantitative methods. The main premise is that qualitative methods should prepare base data for quantitative methods.

The work [3] offers the idea of "methodological triangulation" – an extended model of methods integration. The integration discussed allows receiving information as to the extent the results obtained using different methods agree or disagree. A common limitation in known works is lack of methods compatibility analysis, analysis of integration techniques, scaling of input and output parameters, choice of results aggregation rules, etc.

Consequently, CI safety assessment methods integration must ensure both validity check for results obtained and enhanced assessment validity as a result of maximum coverage of the whole set of input data by a minimum set of methods and information technologies used.

Fuzzy technologies are actively used for CI safety assessment. Thus, for example, in nuclear industry *Fuzzy Logic and Intelligent Technologies intensively are applied for solving fuzzy control problems*, which can't be solved using existing methods and approaches.

BBNs are also widely used in system safety assessment tasks characterized by uncertainty, imperfect knowledge, influence of a variety of random factors. Thus, e.g., BBNs are used as a basis for creation of the expert diagnostics system for NPP operators [4], for modeling complex industrial facilities [5], as well for evaluation of reliability and safety assessment in complex systems [6].

The aim of this chapter is to introduce an approach to series integration of CI safety assessment methods using integration of FL methods and BBNs under uncertainty.

2 Joint FL-BBN Assessment of CI Safety

2.1 General Approach

The suggested approach is based on the following assumptions:

- any CI may be represented as a collection of hierarchical layers of objects, and namely, systems components and elements;
- any object in CI may be represented as a BBN.

CI hierarchy is a basic premise for representation of its safety assessment integration methods architecture as hierarchy as well. This means that parameters of conditions of, e.g., elements are used as input data for components safety assessment. Further these assessments serve as input data for determining subsystems safety. In this way safety assessment runs from the bottom to the top, from systems of the lowest hierarchy layer to systems of higher layer. The safety of a system is a function of safety of its subsystems, components and elements.

On the other hand, subsystem safety assessments may be unitized in prediction (diagnostics) of their components condition. In this case safety assessment runs from the top to the bottom from systems of the highest hierarchy layer to lower layer systems.

Consequently, both upward and downward integration of methods is possible. Such integration of different methods results in compensation of insufficiency of data for models of higher level due to "excessive" data in another, lower hierarchy layer.

The hybrid safety assessment method suggested by this approach is given in Figure 1.

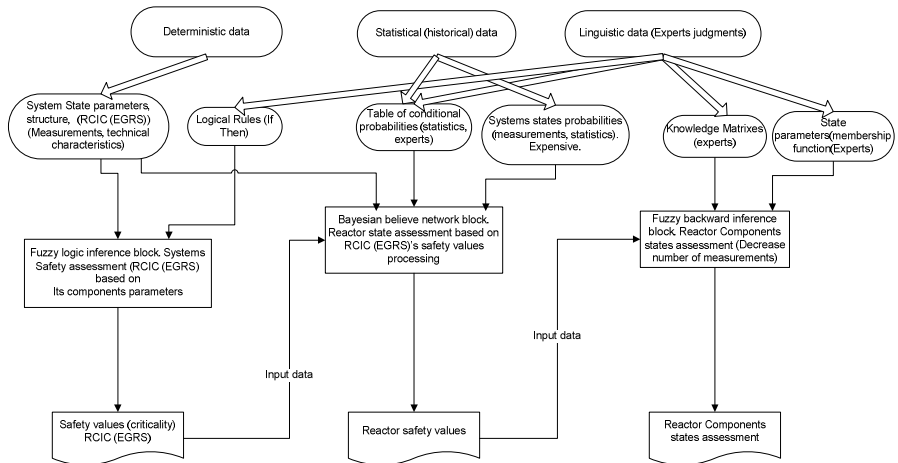


Fig. 1 Hybrid CI safety assessment method

2.2 System Criticality as a Safety Value

A high criticality $Crt(S_i)$ of a system corresponds to its marginal (pre-emergency) state, in which its further use is prohibited or inexpedient or its recovery to operable condition is not possible or expedient. The main distinction between the margin state in reliability theory and high criticality in safety theory is consideration of system failure consequences in pre-emergency condition.

Criticality assessments may be represented on qualitative and quantitative scales. This paper considers linguistic criticality assessments. Thus, for example, criticality can be represented as linguistic variable with terms {High (H), Medium (M), Low (L)}.

An illustration of semantic interpretation of linguistic terms of criticality of condition of, e.g., NPP reactor, is presented in Table 1.

Table 1 Semantic interpretation of linguistic terms of criticality of systems (reactor case study)

Reactor safety levels	Physical State Description
Criticality state – HIGH (reactor emergency state)	Uncontrolled power increase in the reactor core (heat generation), decreased coolant consumption (heat removal) and increased pressure in the primary coolant circuit. Reactor parameters are close to the rated values. For fuel elements these are fuel temperature, cladding temperature, burnout ratio, temperature of physical and chemical processes, heat flow. For the circuit these are pressure, temperature, brittle fracture ratio, pressure differentials
Criticality state – MEDIUM (Reactor pre-emergency state)	The state of unstable equilibrium of the reactor. The reactor is in a state of physical and thermohydraulic stability, which can be upset even by slight disturbances
Criticality state – LOW	Normal routine mode of reactor operation

2.3 Procedure

In order to apply the suggested approach one will need: to chose a test object (system), for which safety rating will be established, the result is determination of the child system for using BBN block; specify which systems define safe state of the test object; the result is determination of the parent system for using BBN block; determine components of the parent system and parameters of their states; the result is logic and linguistic model of the parent system for determining their safety rating in terms of parameters of components for FLI block.

1. Fuzzy logic inference (FLI) block (bottom-up analysis) for CI safety systems assessment on the basis of parameters of its components. In order for the block to solve problems it should have solved the subtask of selecting the most important system components, which condition defines system safety. The task of forming

of a set of *informative (essential) parameters*, the values of which allow distinguishing system conditions, must be solved. The basic data are deterministic input data – parameters of components operation. Output data are criticality condition of the system.

2. BBN block for CI safety assessment. The set of CI systems is divided into two sets: parent and child systems. When using BBN parent systems criticality conditions are used for determining criticality of the child systems. The basic data are parameters of criticality conditions of the parent systems obtained in the FLI block and conditional probability table (CPT). CPT determines the relation between system conditions. Probabilities can be represented on absolute and fuzzy scale. Input data are criticality condition of the child system.

3. Fuzzy backward chaining block to obtain predictive estimates of condition parameters of child system components. Probability distribution of estimates of child system criticality obtained using BBN is used as input data to derive logic equations. Additional information is expert knowledge matrix R. The block's output data is predictive estimates of component conditions.

3 Application of FL-BBN Method for NPP Reactor Safety Assessment

3.1 FLI Block Application

NPP reactor safe condition is a function of a number of systems. Let us focus on the Reactor Core Isolation Cooling (RCIC) System and the Emergency Gas Removal System (EGRS) as an illustrative example. Importance of these systems for safe reactor condition was clearly demonstrated by NPP accidents. Their condition and reliable operation are critical for reactor safety.

RCIC is the first parent system for the reactor in terms of BBN (child system). It is designed for core emergency cooling. It is comprised of three interrelated systems: primary, back-up and continued cooldown subsystems.

EGRS is the second parent system which performs the function of noncondensable gases removal from the first circuit, protects fuel elements, prevents natural circulation failure in the first circuit.

Consider the use of the FLI block to assess the criticality state of the RCIC.

The RCIC safety assessment task is represented as the task to find a representation in the following form:

$$X^* = (x_1^*, x_2^*, x_3^*, \dots, x_n^*) \rightarrow d_j \in D = (d_1, d_2, d_3, \dots, d_m), \quad (1)$$

where X^* – a set of parameters describing the state of RCIC components; D – a set of probable d_j , $j = 1, m$, RCIC safety values.

The first subtask of the block is to choose the set of components that are most important in terms of RCIC core cooling performance. For example, pumps, the condenser can be treated as such components. Reliable operation of any one of the pumps is the critical aspect from the viewpoint of RCIC safety functions.

The second subtask of the block is to select functional parameters $x_1 \div x_n$ that evaluate the states of important RCIC components. Among critical parameters that evaluate pump stare are feed (F), pressure (P), rate of revolution (RR), water reserve in the condenser (C), etc. Increase (decrease) in these parameters with respect to certain values may be an indication of malfunctions or failures resulting in RCIC safety function degradation.

In this way, in order to assess RCIC safety it is necessary:

- to determine values of parameters describing RCIC components functioning

$$X^* = (x_1^*, x_2^*, x_3^*, \dots, x_n^*); \tag{2}$$

- to plot a diagrams of RCIC safety linguistic terms membership function $\mu^{a_i^{jp}}(x_i^*)$;

- to determine values of the membership function $\mu^{a_i^{jp}}(x_i^*)$ at fixed values of parameters $X^* = (x_1^*, x_2^*, x_3^*, \dots, x_n^*)$;

- using logic equations in the following form:

$$\begin{aligned} \mu^{d_m}(x_1, x_2, \dots, x_n) = & \mu^{a_1^{m1}}(x_1) \wedge \mu^{a_2^{m1}}(x_2) \wedge \dots \wedge \mu^{a_n^{m1}}(x_n) \vee \mu^{a_1^{m2}}(x_1) \wedge \dots \\ & \wedge \mu^{a_2^{m2}}(x_2) \wedge \mu^{a_n^{m2}}(x_n) \vee \dots \vee \mu^{a_1^{mk_m}}(x_1) \wedge \mu^{a_2^{mk_m}}(x_2) \wedge \dots \wedge \mu^{a_n^{mk_m}}(x_n), \end{aligned} \tag{3}$$

\vee – logical OR, \wedge – logical AND,

to determine values of membership functions for all possible RCIC safety values.

A knowledge base used to derive logic equations for RCIC is presented in Table 2.

Table 2 RCIC knowledge base

Feed	Pressure	Rate of revolution	Condenser water reserve	RCIC criticality values
L	L	L	L	H
L	M	L	L	H
L	M	M	L	H
L	M	M	M	M
.....				
H	H	M	H	L

Within the scope of the example the logic equations are of the form:

$$\begin{aligned} \mu_{y_i}(Crt = High) &= [0.12 \wedge 0.55 \wedge 0.7 \wedge 0.66] \vee \\ & [0.12 \wedge 1.0 \wedge 0.7 \wedge 0.66] \vee [0.12 \wedge 1.0 \wedge 0.87 \wedge 0.66]=0.12; \\ \mu_{y_i}(Crt = Medium) &= [0.12 \wedge 1.0 \wedge 0.87 \wedge 0.91] \vee [0.87 \wedge 0.55 \wedge 0.87 \\ & \wedge 0.66] \vee [0.94 \wedge 0.55 \wedge 0.87 \wedge 0.91]=0.55; \\ \mu_{y_i}(Crt = Low) &= [0.87 \wedge 1.0 \wedge 0.87 \wedge 0.91] \vee [0.94 \wedge 0.66 \wedge 0.53 \wedge 0.55] \\ & \vee [0.94 \wedge 0.66 \wedge 0.53 \wedge 0.91]=0.87. \end{aligned}$$

Select d_j^* as a solution, for which,

$$\mu^{d_t^*}(x_1, x_2, \dots, x_n) = \max[\mu^{d_j}(x_1, x_2, \dots, x_n)], j = \overline{1, m}, t = \overline{1, m}. \tag{4}$$

Criticality state of EGRS is determined in a similar manner. EGRS safety estimates are determined in the FLI block in terms of parameters of its components (excess steam-gas mixture removal bypass conduit, bypass conduit steam-gas mixture signal indicator, steam-gas pressure chamber).

3.2 BBN Block Application

The complex of systems including the reactor (child system) and RCIC and EGRS (parent systems) can be represented in the form of BBN.

This approach uses BBN for:

- reactor criticality condition prediction according to the state of parent systems (RCIC and EGRS). This involves recalculation of probability of the reactor child system being in each of its possible criticality conditions depending on incoming BBN parent systems condition change evidence using the CPT;
- determination of conditions of the parent systems (RCIC and EGRS) according to evidences (facts) of their possible condition (diagnostics task).

In BBN probabilities of the reactor being (S_3) in different conditions of set S_3 depending on conditions of the parent systems (RCIC- S_1 , EGRS- S_2) can be determined by the relation of the following form:

$$P(S_3^{(k)}) = \sum_i \sum_j P(S_3^{(k)} / S_1^{(i)}, S_2^{(j)}) * P(S_1^{(i)}) * P(S_2^{(j)}) \tag{5}$$

where $P(S_3^{(k)})$ - probability of S_3 being in k -th condition;

$P(S_3^{(k)} / S_1^{(i)}, S_2^{(j)})$ - conditional probability of S_3 system in k -th condition given that S_1 system is in i - th condition and S_2 system in j - th condition. Conditional probabilities for BBN are set by professional expert;

$P(S_1^{(i)})(P(S_2^{(j)}))$ - probability of $S_1(S_2)$ system being in i -th (j -th) condition.

This approach predicts reactor safety state without additional measurements of reactor parameters (pressure, temperature, etc.).

The block's output data are reactor safety predictive estimate represented in the form of the following probability distribution:

$$P(\text{Crt}(R) = \text{High}) = 0,6; P(\text{Crt}(R) = \text{Medium}) = 0,3; P(\text{Crt}(R) = \text{Low}) = 0,1.$$

The choice of reactor safety assessment is made according to *the maximum probability criterion*. Reactor condition assessment is a complex and costly task. Fukushima-1 accident proved the importance of the need for reliable operation of monitoring systems (detectors). These risks can lead to a situation, when the operator can completely lose all the sense of what is happening to the reactor. For support and decision making in case of station black-out it is necessary to use all the information (including indirect information) for reactor state assessment. Consequently, it is important to solve the problem of predicting reactor components condition without any measurement and using additional information. This task is solved in the fuzzy backward chaining block. Primary importance of fuzzy backward chaining is that it considers parameters, which are essential for safety thought physical measurement of which is substantially limited.

3.3 Fuzzy Backward Chaining Block for Prediction of Reactor Components Condition Parameters

The problem of fuzzy backward chaining lies in evaluation of input parameters describing reactor components condition provided that the matrix of knowledge and reactor safety and output estimations are known.

In terms of input A and output B sets link between them can be represented in the following form $B = A \circ R$, where A(B) - a fuzzy set of input (output) parameters specified in space X(Y).

Matrix of knowledge R can be represented as

$$M_p = \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ r_{21} & r_{22} & \dots & r_{2n} \\ \dots & \dots & r_{ij} & \dots \\ r_{m1} & r_{m2} & \dots & r_{mn} \end{pmatrix},$$

where r_{ij} – an element of matrix expressing the level of confidence of the expert in existence of cause-and-effect relations between component input parameter and corresponding output parameter describing safety of the system.

Considering BBN block the following distribution was obtained:

$$P(\text{Crt}(R) = \text{High}) = 0,7; P(\text{Crt}(R) = \text{Medium}) = 0,1; P(\text{Crt}(R) = \text{Low}) = 0,2.$$

Introduce parameters conditions vector y_1, y_2, y_3 . These parameters are values of the vector of criticality probability distribution (BBN output parameters).

State this vector in the following form:

$$B = 0,7|y_1 + 0,1|y_2 + 0,2|y_2$$

In the fuzzy backward chaining block the expression, for example, $0,1|y_2$ means that the level of the expert's confidence that the system is in a certain condition $\text{Cr}(S_1) = \text{Medium}$ is equal to 0.1.

It is necessary to find such a fuzzy set $A = \{\mu(x_1)|x_1, \mu(x_2)|x_2, \dots, \mu(x_n)|x_n\}$, that would correspond to fuzzy set B. Fuzzy set A can be represented as a vector $a = (a_1, a_2, \dots, a_n)$, where a_n – corresponding value of membership degree $\mu(x_n)$ of the reactor components condition parameter.

In this example reactor components are fuel elements S_{11} and circuit S_{12} . Considered are two parameters a_1 – fuel element temperature and a_2 – pressure in the circuit.

Examination results are presented as knowledge matrix in the following form:

$$R = \begin{vmatrix} 0,9 & 0,1 & 0,2 \\ 0,6 & 0,5 & 0,5 \end{vmatrix}$$

Considering the knowledge matrix and probability distribution in view of BBN the following logic equation was produced:

$$[0,7 \quad 0,1 \quad 0,2] = [a_1 \ a_2] \circ \begin{vmatrix} 0,9 & 0,1 & 0,2 \\ 0,6 & 0,7 & 0,5 \end{vmatrix}$$

When using max-min compositions the latter relation rearranges to the following form:

$$0,7 = (0,9 \wedge a_1) \vee (0,6 \wedge a_2)$$

$$0,1 = (0,1 \wedge a_1) \vee (0,7 \wedge a_2)$$

$$0,2 = (0,2 \wedge a_1) \vee (0,5 \wedge a_2)$$

Solution of this equation produces the following values: $a_1 = 0,7; 0 \leq a_2 \leq 0,1$.

In this way, reactor condition obtained using BBN is influenced by high temperature of fuel elements since it is this premise that corresponds to the highest value of membership function.

4 Conclusions

The chapter considers an approach to series integration of safety assessment methods. Two integration architecture types are introduced: series integration and parallel. The series integration might be useful to increase the safety values' validity. The parallel integration allows reduce the amount the safety – related information. Integration of BBN and FL allows capturing all information available

information required for safety assessment of complex dynamic system under uncertainties. Application of FL methods when all parameters describing the system operation are known allows determining the criticalities of all systems under interests. But for complex dynamical systems the processes of parameters' measurement might technically difficult. Application of BBN allows decreasing the amount information. Thus, for example, for FL-based safety assessment of reactor, RCIC and EGRS it is required to measure all parameters of all systems. Integration of FL-based methods and BBNs allows decreasing the amount of input information (measurements) and not measure reactor parameters. RCIC and EGRS parameters are only required. Thus in the scope of example this integration decrease on tierce of required information. This approach might be considered as a basis for the expert system to help the operator make the decisions when I&C ability to measure the critical parameters is compromised due to NPP blackout.

References

- [1] Leech, N., Onwuegbuzie, A.: Qualitative data analysis: A compendium of techniques for school psychology research and beyond. *School Psychology Quarterly* 23, 587–604 (2008)
- [2] Prein, E., Kelle, U.: Strategien zur Integration qualitativer und quantitativer Auswertungs- verfahren, p. 234. Universitat Bremen (1993)
- [3] Johnson, B., Christensen, L.: Educational research: Quantitative, qualitative, and mixed approaches, p. 34. Sage Publications, Thousand Oaks (2008)
- [4] Kang, C., Golay, M.: Bayesian belief network-based advisory system for operational availability focused diagnosis of complex nuclear power systems. *Expert Systems with Applications* 17(1), 21–32 (1999)
- [5] Weber, P., Jouffe, L.: Complex system reliability modeling with Dynamic Object Oriented Bayesian Networks (DOOBN). *Reliability Engineering and System Safety* 91(2), 149–162 (2006)
- [6] Weber, P., Lung, B.: System approach-based Bayesian Network to aid maintenance of manufacturing process. In: 6th IFAC Symposium on Cost Oriented Automation, Low Cost Automation, Berlin, Germany, October 8-9, pp. 33–39 (2001)
- [7] Brezhnev, E., Kharchenko, V., et al.: Dynamical and Hierarchical Criticality Matrixes-Based Analysis of Power Grid Safety. In: Proceeding of ANS PSA 2011 International Topical Meeting on Probabilistic Safety Assessment and Analysis, Wilmington, NC, March 13-17, pp. 1137–1149 (2011)

Towards Evolution Methodology for Service-Oriented Systems

Szymon Kijas and Andrzej Zalewski

Warsaw University of Technology,
Institute of Control and Computation Engineering, Warsaw, Poland
s.kijas@elka.pw.edu.pl,
a.zalewski@ia.pw.edu.pl

Abstract. Modern organisations are forced to evolve their IT systems to keep up with ever-changing business requirements. Service-Oriented Architecture addresses the challenge of boosting a system's modifiability by composing a new functionality out of existing, independent, loosely-coupled services. This makes SOA a promising design paradigm for rapidly evolving systems. However, existing development methodologies for SOA, such as IBM's SOMA, focus more on the transition from legacy non-SOA to SOA systems, and less on their subsequent evolution. This makes the development of an evolution methodology suitable for service-oriented systems an open research problem. The presented evolution methodology comprises an evolution process and an evolution documentation model. The process is compliant with a popular ISO 20000 norm. Its artefacts have been defined in terms of the evolution documentation model. The business-driven changes are documented with architectural decisions that capture changes made to the system at various levels of scope, together with their motivation. In order to facilitate the change-making process, a set of typical change scenarios has been defined. It comprises typical sequences of architectural decisions for cases of the most important changes. The entire approach is illustrated with a real-world example of an internet payment system.

1 Introduction

A system's modifiability is a primary concern for many modern organisations, which are striving to evolve their system's to meet frequently changing or emerging business requirements. Service-oriented architectures support a system's modifiability by enabling the development of a new functionality by a loose, easy-to-modify composition of existing services into new ones and by the extensive reuse of already existing services. This makes service-oriented architectures a design paradigm, which is particularly suitable for intensively evolving systems.

However, neither SOA development methodologies such as SOMA [6], SOMF [7], nor existing traceability methods [20], provide efficient and complete support for the evolution of SOA systems. This reveals a gap between the real needs of

SOA adopters and existing SOA development methodologies. This motivates our research on a methodology for evolving service-oriented systems presented and discussed in this paper.

The rest of the paper has been organised as follows: related work is briefly discussed in section 2, the evolution methodology is presented in section 3, its application has been illustrated on a real-world example in section 3, the contribution of this paper is discussed against the related work in section 4, and finally the outcomes and further research outlook is presented in section 5.

2 Related Work

The evolution of service-oriented systems is quite a new area of research, with rather a sparse publication record as the envisaged service-oriented world, in which services composition is the primary means of developing new functionality, is a world to come rather than the world we actually live in. In practice, service-oriented systems are currently built on top of already existing non-service-oriented ones for integration purposes [1]. Therefore, a lot of research effort has been devoted to addressing the issue of migrating legacy systems to SOA. Suitable methods can be found in, for example [6], [7], [8], [9], [10], [13]. So far, the identification and development of services to “wrap” existing functionality into services and enable interaction between systems has been the main research focus. The evolution was understood as making changes to the services, i.e. their interfaces, functionality, etc. (compare, for example [6]). The emergence of a market for third-party services and the deployment of more systems crossing organisational boundaries, possibly making their services publicly available, will change the above condition and make the evolution of business processes and service compositions a primary focus.

The research record on the maintenance and evolution of service-oriented systems is rather sparse. An idea of a transformation-driven method for evolving service-oriented systems has been sketched in [22], which seems to be, so far, the only development of this kind. Most of the research carried out so far only concerned selected evolution issues, such as changes traceability [14] (a framework for tracing changes between models of service-oriented systems), change propagation [16], [19], versioning [15], impact analysis [17], model-driven approaches to service composition, for example [18]. The research challenges in this field have been investigated in papers [1], [2], [3]. Paper [1] indicates that the development of maintenance processes is still an open research issue. Nevertheless, maintenance has been included in a post-deployment phase in [11], and has been provisioned for in the methodology presented in [12]. The evolution of services has been accounted for in the fractal process of SOMA methodology with the concept of successive iterations. In [18], the authors propose to use change management mechanisms to control the evolution of service compositions. An extensive framework for capturing architectural decisions comprising a SOA system design has been presented in [21].

Finally, let us observe that ISO 20000/ITIL [4], [5] is a set of practices for change management that has been widely accepted and adopted in industrial practice.

3 Evolution Methodology for Service-Oriented Systems

Software development methodologies, such as object-oriented or structured ones, have traditionally comprised two basic components: the development process, being a kind of a design recipe, and the supporting tools, which are used within the development process (models, notations, modelling and model analysis techniques). Our evolution methodology for service-oriented systems follows this scheme and comprises:

- **Evolution process** – defines a workflow, which defines how the modifications requested by business should be done in a disciplined, repeatable way, which is compliant with established industrial standards (ISO 20000/ITIL) – section 3.1;
- **Evolution supporting tools** – includes models used to capture the evolution of the SOA system, i.e., the model of the SOA system together with evolution documentation model, as well as techniques supporting the development of changes. The latter include: change scenarios, enriched traceability mechanism, impact analysis technique – see section 3.2.

3.1 Evolution Process

The evolution process defines a disciplined and controllable way of making numerous changes to the system. The evolution process comprises a set of instances of the modification process (fig. 1), which are initiated for every submitted Request for Change document (RFC). The modification process consists of four basic phases, which are compliant with the change management process defined in the ISO 20000:2005 standard:

- **Change assessment** – requested changes, described in the RFC, are assessed in terms of their impact (on quality attributes, SLAs, other processes, services, etc.), urgency, cost, benefits and risks;
- **Change approval** – decision makers accept or reject the submitted change. This decision is based mainly on business factors. Subsequently, the development of the approved changes is scheduled.
- **Change development and deployment** is a configurable part of the Modification process; various development processes can be applied here, e.g., agile Feature Driven Development, Scrum, XP or non-agile: waterfall, RUP. The choice should depend on the established development practices and experience of the development team.

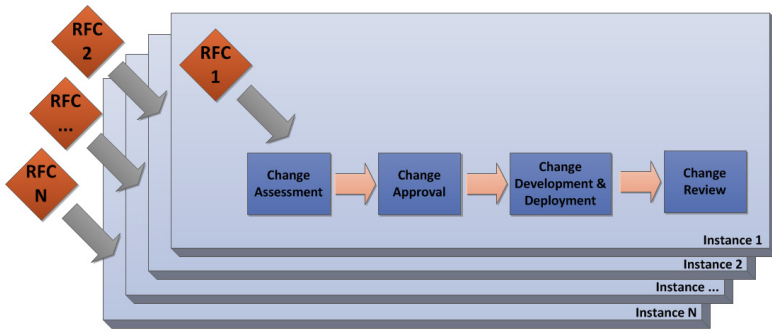


Fig. 1 Overall Structure of the Evolution Process

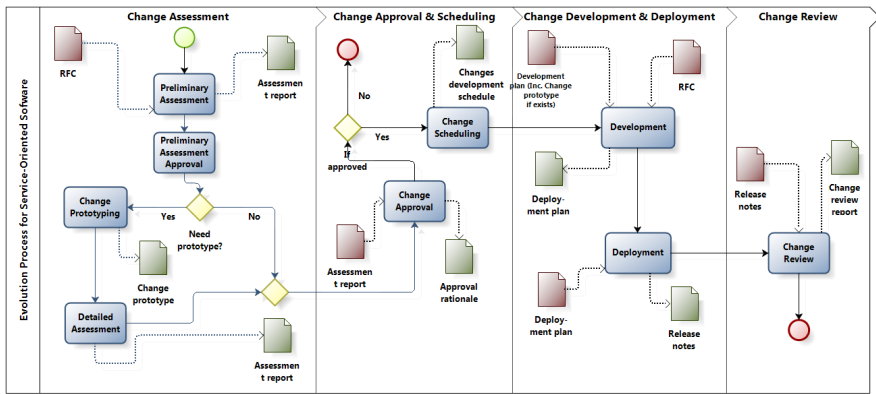


Fig. 2 Detailed workflow of the evolution process for service-oriented systems

- **Change review** is an optional phase, as required by ISO 20000. However, it should be defined whether organisation wants to include the reviews of deployed changes in its change management practices.

The detailed workflow of the modification process has been shown in fig. 2. The “Change Assessment” phase starts from a “Preliminary Assessment”, in which changes described in the RFC are assessed on the basis of expert knowledge of business and system analysts in terms of their impact on functionality, quality (including Service Level Agreements), the effort needed to complete the changes and risks connected with implementing and deploying the change. The results of such an assessment are examined in a “Preliminary Assessment Approval” task, in order to verify whether they are sufficiently credible and complete in order to decide about the acceptance or rejection of the change.

“Change Prototyping” is performed if more detailed information on the impact of a change is needed in order to assess the requested change. A change prototype is a partially developed model of changes (compare section 3.2) that is supposed

to facilitate an in-depth impact analysis and will become a basis for further development if the change is approved. The final decision about the approval or rejection of changes takes place in “Change Approval and Scheduling” phase. Approved changes have to be appropriately scheduled (“Change Scheduling”) to avoid conflicting changes being developed at the same time. This may also result in combining two or more changes to be developed as a single chunk. The rest of the modification process workflow seems to be self-explanatory. The artefacts of the evolution process have been defined in section 3.3.

3.2 Evolution Documentation Model

The Evolution Documentation Model consists of two basic components:

- SOA System Model (section 3.2.1) – a set of models representing the components of service-oriented systems (business processes, services, service operations and their internal logic, service compositions) at various levels of detail;
- Evolution Capturing Model (section 3.2.2) – documenting the changes introduced by the evolution steps. Such changes may concern every artefact of the SOA System Model. The evolution model provides a traceability mechanism for SOA System Models, and also facilitates impact analysis and capturing the architectural knowledge emerging during the development of changes.

3.2.1 SOA System Model

Service-oriented systems, such as presented in section III, implement one or more business processes, whose activities are supported by suitable business services. These services, in turn, comprise a number of service operations. These may be associated with the composition of a service (composed of other service operations) or developed source code. These dependencies have been reflected in the SOA System Model (fig. 3), which comprises the three layers described beneath.

Business Process Layer consists of a set of “Business Processes” supported by a service-oriented system. These BPMN models abstract from the implementation details such as service compositions, services definitions, interfaces, operations, operations’ arguments, etc. Each business process is associated with a set of tasks (class “Task”), which are also included in the workflow represented in BPMN. “Task” is described using: name and description, and optionally: input and output documents (denoted by an associations with “Document” class). “Document” is described by its name and optionally: description and/or state.

Service Layer comprises a set of models that represent services used to support business processes. These models form a cascading, recursive structure as a model of a service is connected with a number of service operations, each of which can

be either an invocation of a basic (non-composed) service operation, or of a service composition, etc. The Service Layer comprises the following classes:

- **“Service”** consists of: name, set of service operations (represented as associations with “Service Operation”). Therefore, service is rather a kind of a container, or just a label for the set of its operations.
- **“Service Operation”** is an entity in which computation actually takes place. This class contains: operation name and input document – the document fed into the operation or/and output document that is the outcome of the computations (expressed as an association to “SOA Document” class).
- **“Service Composition”**: model in BPMN that expresses the workflow composed of the invocations of service operations (service operations belonging to various services – internal and provided by the external providers). Service composition should be assigned to the service operation that actually provides its input and output interface.
- **“SOA Document”** contains: the name of the document and the structure of its content (i.e. XML, text or binary data). Such a document should correspond to a single “Document” from the business process layer.

Low Level Models Layer – low level, detailed models (typically in UML) and executable code. Note that these models may concern only basic services developed in-house, or being in the possession of the system’s owner.

It is worth emphasising that the SOA System Model reflects the structure of real world service-oriented systems, which is particularly noticeable in the relation between services and their operations. We also assume that the tasks can be one-to-one associated with service-operations, which implement them in a service-oriented system. The same applies to the Documents and SOA Documents. This imposes certain rigour both on business analysts and SOA system designers, which is needed to make business even closer to IT.

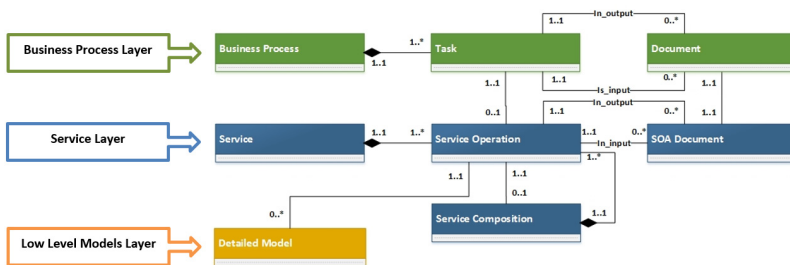


Fig. 3 Detailed architecture model of SOA system

Example. Evolution of an Internet payment system

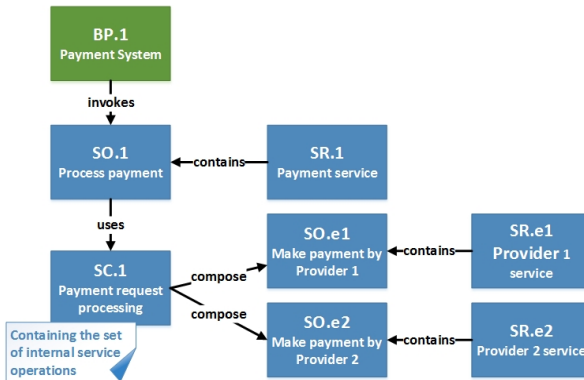


Fig. 4 SOA System Model for the “Payment System”

All the components of the Evolution Documentation Model have been illustrated on a real-world example of a portal supporting internet payments (named “Payment System”). The system comprised among others: web portal for individual customers’ payments, web module for system administration and service dedicated for mass payment customers – named as “Payment service”. The initial version of the system supported only two payment methods: credit or debit card payments and wire transfer payments.

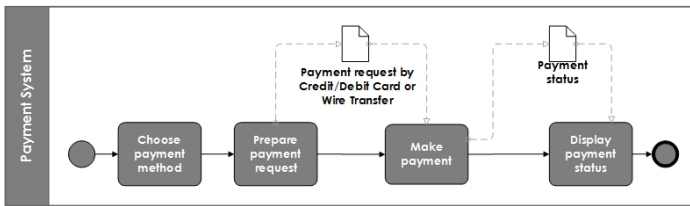


Fig. 5 Business process “Payment System”

The SOA System Model of the Payment System has been presented in fig. 4 (documents have been omitted for the clarity of the picture). It contains:

- Business Processes “Payment system” (BP.1, fig. 4): the model of a payment process implemented by the portal (fig. 5).
- Services: “Payment service” (SR.1), external services: “Provider 1 service” (SR.e1) and “Provider 1 service” (SR.e2).
- Service operations:
 - o “Process payment” (SO.1), which accepts “Payment request” document and after processing the “Payment status” document is returned);

- o Internal and external (“Make payment by provider 1” (SO.e1) and “Make payment by provider 2” (SO.e2)) services’ operations invoked inside service composition described below.
- Service compositions: BPMN model of service composition “Payment request processing” (SC.1), which is assigned to “Process payment” service operation. It has been composed out of several service operations provided internally or externally (compare fig. 6).

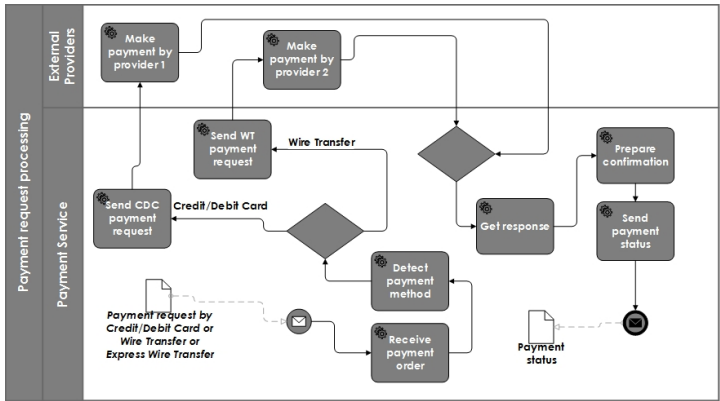


Fig. 6 Service composition “Process payment request”

The business process in fig. 5 is the “macro-flow” of the Payment system, while service composition “Payment Request Processing” defines (fig. 6) the “micro-flow” of payment processing.

3.2.2 Evolution Capturing Model

The Evolution Capturing Model (fig. 7) documents evolution as a set of “Evolution Steps”. Each Evolution Step is triggered by RFC document (Request For Change), which specifies the requested change, describes its motivation, business, and if needed, technical context. The step itself comprises a cascade of architectural decisions, which capture the changes made to the models of different levels of SOA System Model. The changes made to a service-oriented system are of a cascading structure, i.e., change to a business process may force changes to services, these in turn may force changes to service compositions, which in turn may require changes to services etc. Such a cascading effect is reflected by “forces” associations.

A set of typical modification scenarios has been developed in order to facilitate the development of changes (table 1). Let us note that the change scenarios can be applied recursively.

Architectural decisions connect previous (is_input association), modified versions resulting from change’s implementation (is_outcome association) as well as models’ alternatives considered during change’s development (is_alternative association). At the same time they provide rationale for the changes made, e.g., by justifying the choice between the connected alternatives. This concerns the following components of SOA System Model: Business Process Models, Service Models, Service Composition Models, Detailed Models.

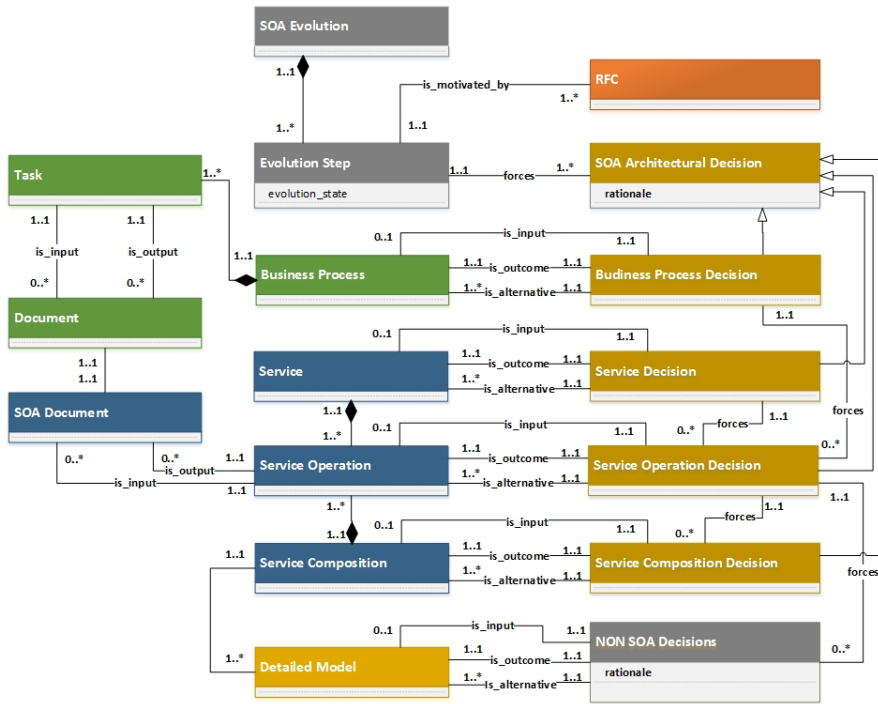


Fig. 7 Evolution Capturing Model

Example. Evolution of Internet Payment System (cont.)

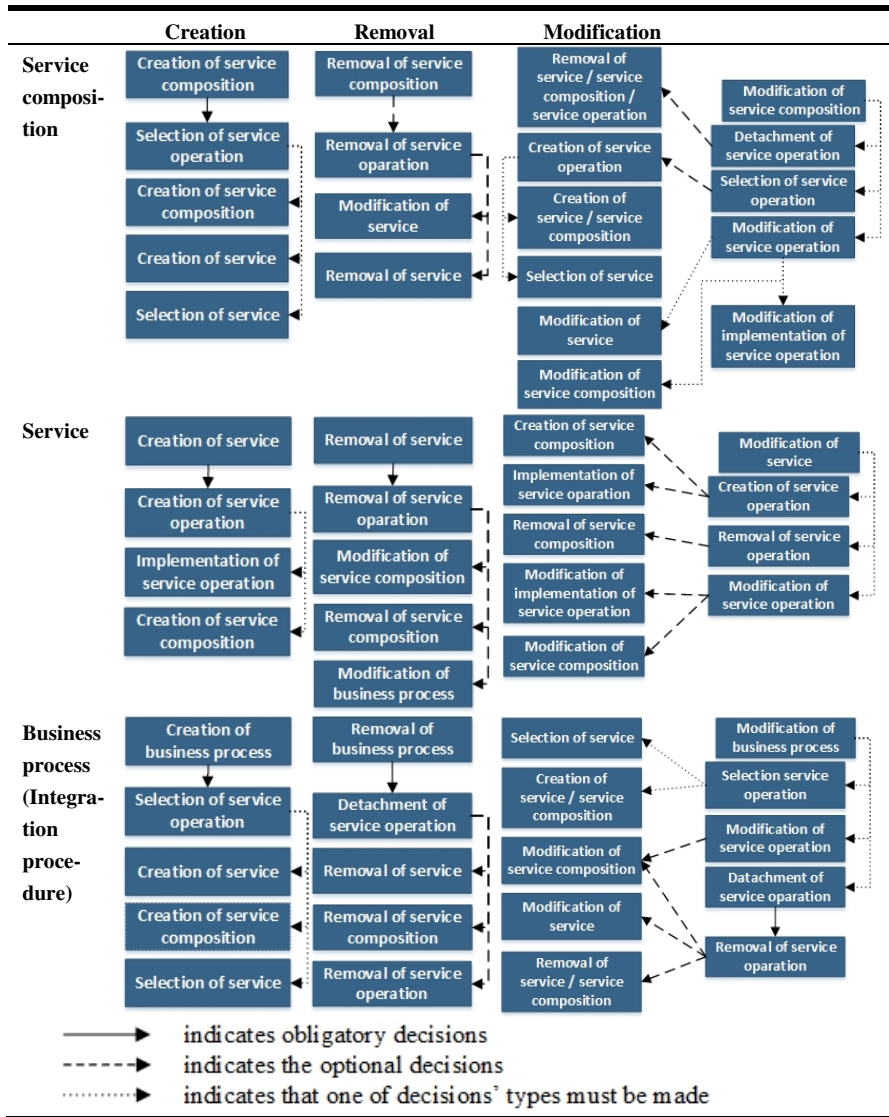
Let us look back at the example to see how changes are captured using the Evolution Documentation Model presented in fig. 7.

Evolution Step No 1

Summary of RFC Document: The business expects that instant wire transfers (normally transfers are made during several communication sessions a day) will also be available.

The cascading changes necessary to implement the modifications described in RFC have been illustrated in fig. 8. The sequence of modification scenarios applied in order to develop the changes depicted in fig. 8 has been shown in fig. 9.

Table 1 The set of most popular SOA decision-making scenarios



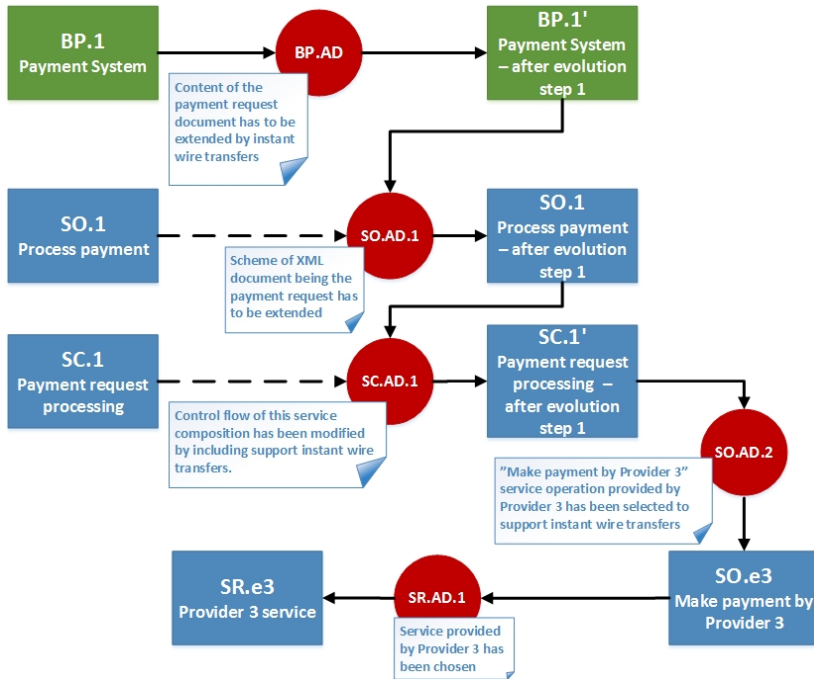


Fig. 8 The first evolution step of the payment system

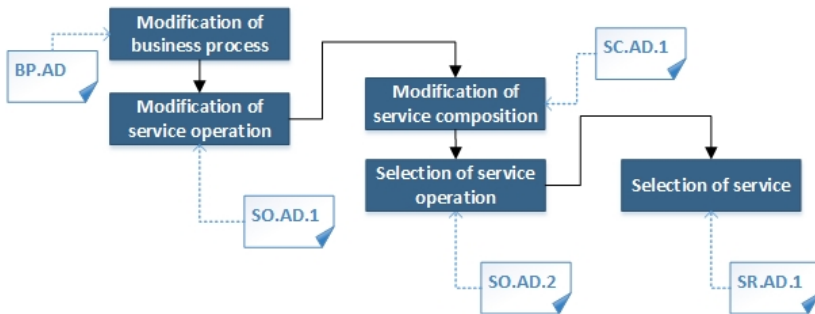


Fig. 9 Sequence of change scenarios applied to modify the system in order to support instant wire transfers

The following artefacts had to be modified:

- Business process “Payment System” – its control flow (fig. 5) remained unchanged, though, the content of the “payment order” document has been extended to include data necessary to issue an instant wire transfer.
- Service operations:

- Service operation “Process Payment” has been modified in order to support instant wire transfers – the XML scheme of the “Payment request” SOA document (corresponding to the workflow’s “Payment request” document) has been extended with the information necessary for the instant wire transfers.
- Service operation “Make payment by provider 3” (SO.e3) has been added and invoked in the service composition “Process payment request”.
- Services – service “Provider 3 service” (SR.e3) was added, which contains operation supporting instant transfers;
- Service composition “Process payment request” has been extended with the invocation of the service operation “Make payment by provider 3” supporting instant wire transfers (fig. 10). The composition’s workflow was appropriately adjusted.

Evolution Step No. 2

Summary of RFC Document: The business expects that international instant wire transfers will also be available.

Implementation of the above changes required that a cascade of architectural decisions had to be made. These decisions capture the changes made to the models of “Internet Payment System” and their rationale. This decision making process has been illustrated in fig. 11, which extends the model developed in order to capture changes made in step No. 1. Modified versions of business process “Payment System” and service compositions “International payment request processing” can be found in fig. 12 and 13, respectively.

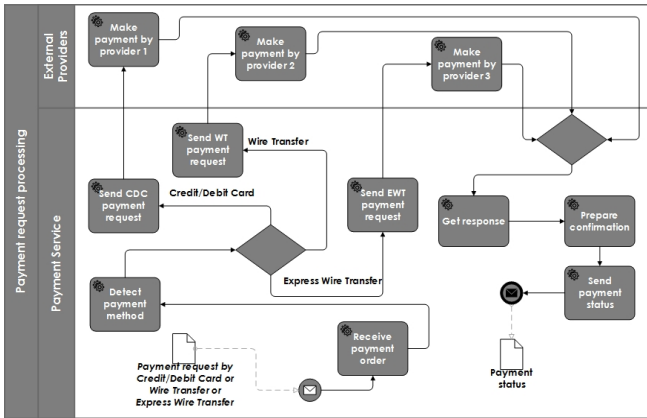


Fig. 10 Service composition “Process payment request” service operation after the first evolution step

3.3 Evolution Supporting Techniques

The above example illustrates how Evolution Capturing Model can be employed so as to document the changes made to the system. Let us note that every architectural decision can not only define a traceability link between two consecutive versions of a certain model but can also be connected with the considered model's alternatives. Architectural decision includes also modification's rationale, which explains why certain changes have been made to the system. This makes it possible to understand how system has reached its current shape.

The structure of SOA System Model enables top-down impact analysis as the models potentially affected by the changes can be discovered by following the associations between higher- and lower-level (more detailed) models. The set of potentially affected models tightens as more detailed decisions are made. In such case the top-down traceability, goes from already affected models down to the possible affected subcomponents. This way the scope of changes necessary to implement a given change can be established, which should facilitate time and cost estimation. Obviously, there is a lot of space for further research in this area, which was discussed in section 4.

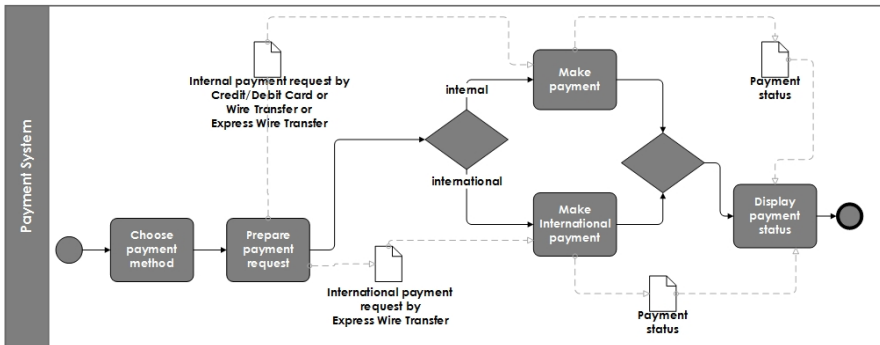


Fig. 12 Business process model “Payment system” after the second evolution step

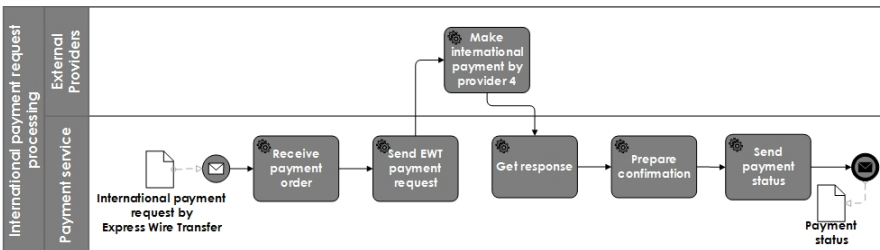


Fig. 13 Service composition “International payment request processing”

3.4 Artefacts of the Evolution Process

A detailed description of all the artefacts produced during the evolution process for service-oriented systems has been summarised in table 2.

Table 2 List of artefacts produced during the evolution process

Artefact	Description	How the Evolution Capturing Model (ECM) and supports the artefacts of the evolution process artefacts
RFC	The change is described in business or technical terms. The document also contains an explanation of the change and indications concerning its importance/priority.	RFC is included in ECM (class RFC).
Assessment report [Preliminary] or [Full]	<p>The document includes:</p> <ul style="list-style-type: none"> • scope of change: <ul style="list-style-type: none"> ○ list of business processes / service operations / service compositions modified/added/removed; • impact analysis – description of a change’s impact on: <ul style="list-style-type: none"> ○ quality (including SLAs), e.g. reliability, performance, business continuity, etc.; ○ list of business processes affected by the changes (e.g. requiring revision); ○ overlapping changes; • cost estimates, • identified risks, • attachments (other documents used for or created during the assessment process), in the case of the [Full] version of the document – change prototypes are included here. 	<p>The scope of a change can be expressed as a set of the instances of classes (Business Process, Service, Service Operation , etc.) of an SOA System Model that are subject to changes.</p> <p>The associations in an SOA System Model enable the impact of changes to be assessed (section 3.2.1) by identifying the artefacts that may require changes.</p>
Change prototype	<p>Set of business process and service composition models containing:</p> <ul style="list-style-type: none"> • modified versions of existing business processes, services and service operations with the associated service compositions, • models of new processes introduced, • list of removed business processes, service compositions, service operations and services • list of detailed models subjected to change / modification / removal <p>The above models are drafts of the assessed changes. They have not been fully developed, verified or tested.</p>	<p>The association <i>is_alternative</i> of ECM indicate the variants of models considered as a possible solution needed to develop a certain change. Chosen (on trial) alternatives of every modified artefact of the model comprise change prototype.</p>

Table 2 (continued)

Change acceptance report	The document contains: <ul style="list-style-type: none"> • notes explaining the need and rationale for the approved change, • effort / cost estimated, • allocation of the cost within budget (the source of change financing); • time schedule for change development and deployment; • attachments including: RFC, Assessment reports and Change prototype. 	ECM enables an analysis of the rationale of changes made to accomplish every evolution step.
Changes development schedule	A document with a schedule of all changes that have to be implemented.	The components of ECM identified as being subject to change can be used as a basis for developing a change's schedule.
Development plan	The document contains all of the information directly connected to the modification of the system: about business process models, service composition models and service models. The development plan contains the system prototype (if one exists). Additionally, this document contains all the information about detailed models and, of course, a complete set of the architectural decisions that have been made.	---
Deployment plan	The deployment plan contains installation/deployment instructions for a new release.	---
Release notes	Report on the deployment containing a list of bugs that have been corrected or are not in the developed version.	---
Change review report	Defined individually by the organisation.	---

4 Discussion

The overarching goal of our research was to develop an approach to the evolution of a service-oriented system that could be easily adopted by industry. This explains our devotion to the compliance of ISO 20000/ITIL. This is naturally an advantage of the proposed solution over the one presented in [22]. This also applies to the approaches for maintenance suggested in [6], [11], [12].

The Evolution Documentation Model provides a three dimensional traceability:

1. The history of changes made to the components of an SOA System Model (business processes, services, services' operation etc.) is captured with architectural decisions linking previous and modified versions of certain models;

2. Architectural decisions enable the motivation of changes made to the system to be captured at various levels of detail;
3. Logic of a change's development is captured with "forces" association linking changes made at various levels of detail.

The above traceability mechanism is compliant with a reference model proposed in [20], i.e. comprising satisfaction links (association between RFC and evolution step classes), evolution links (is_input and is_outcome associations between consecutive model versions and architectural decisions), rationale links (provided by architectural decisions) and dependency links (associations between the classes of the SOA System Model). In [14], the authors present a method for automatic tracing changes between models of SOA systems, both vertically (between more and less detailed models) and horizontally (between models at the same level of detail). We perceive evolution as a process of making intentional changes to the system. Admittedly, it can be facilitated with automated tools, though they cannot eliminate a conscious decision-maker – architect.

The idea of exploiting the advantages of architectural decisions for SOA systems and their evolution is becoming more and more popular. A comprehensive framework for architectural-decision making was presented in [21]. However, it does not account for the evolution of SOA systems and focuses on architectural decisions only, ignoring typical models used for SOA systems and their interrelations. Its intrinsic complexity makes it difficult to comprehend by practitioners, who have rather little time for learning elaborate methodologies. This observation became a foundation for our earlier work [23]. The proposed structure of the Evolution Capturing Model allows MAD to be employed, as a number of alternatives are associated with the architectural decisions documenting the internal logic of a single evolution step.

5 Summary and Outlook

A methodology for evolving service-oriented systems has been proposed. It comprises a disciplined evolution process and a set of models and other tools supporting the development of changes. The models have been validated on a real world example. The process's compliance with industrial standard ISO 20000 should facilitate the application of the presented approach in practice. There are obviously some missing parts of the methodology, which should become the subject of further research. Therefore, the research outlook includes:

- The development of a quality model and methods for analysing how changes impact the quality attributes;
- Supporting the development of changes with predefined model transformations applied in order to ensure that service compositions meet the quality requirements;
- The development of a software tool supporting the methodology;
- Carrying out further and more extensive validation.

Acknowledgement. This work was sponsored by the Polish Ministry of Science and Higher Education under grant number 5321/B/T02/2010/39.

References

- [1] Lewis, G.A., Smith, D.B., Kontogiannis, K.: A Research Agenda for Service-Oriented Architecture (SOA): Maintenance and Evolution of Service-Oriented Systems. Technical Note, CMU/SEI-2010-TN-003 (March 2010)
- [2] Lewis, G.A., Smith, D.B.: Service-Oriented Architecture and its Implications for Software Maintenance and Evolution. In: FoSM 2008, pp. s. 1–s. 10. IEEE (October 2008)
- [3] Kontogiannis, K., Lewis, G.A., Smith, D.B.: The Landscape of Service-Oriented Systems: A Research Perspective for Maintenance and Reengineering. SEI (2007)
- [4] ISO/IEC 20000-1:2005 and 20000-2:2005, Information technology Service management, ISO 20000-1: Specification. ISO 20000-2. Code of practice. ISO/IEC (2005)
- [5] Office of Government Commerce. ITIL V3 Foundation Handbook. The Stationery Office, Updated edition (June 2009) ISBN: 978-0113311972
- [6] Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Ganapathy, S., Holley, K.: SOMA: A method for developing service-oriented solutions. IBM Systems Journal 47(3), s. 377–s. 396 (2008)
- [7] Bell, M.: Service-Oriented Modeling: Service Analysis, Design, and Architecture. Wiley Publishing (February 2008)
- [8] Winter, A., Ziemann, J.: Model-Based Migration to Service-Oriented Architectures. In: Proceedings of the International Workshop on SOA Maintenance Evolution (SOAM 2007), 11th European Conference on Software Maintenance and Reengineering (CSMR 2007), Amsterdam, March 20-23. IEEE Computer Society (2007)
- [9] Lewis, G., Morris, E.J., Smith, D.B., Simanta, S.: SMART: Analyzing the Reuse Potential of Legacy Components in a Service-Oriented Architecture Environment. CMU/SEI-2008-TN-008, Software Engineering Institute, Carnegie Mellon University (2008)
- [10] Ziemann, J., Leyking, K., Kahl, T., Werth, D.: SOA Development Based on Enterprise Models and Existing IT Systems. In: Cunningham, P. (ed.) Exploiting the Knowledge Economy: Issues, Applications and Case Studies. IOS Press (2006)
- [11] High Jr., R., Kinder, K., Graham, S.: IBMs SOA Foundation: An Architectural Introduction and Overview (November 2005)
- [12] Mittal, K.: Build Your SOA, Part 1: Maturity and Methodology. IBM (May 2005)
- [13] Erl, T.: SOA Design Patterns. Prentice Hall (2009) ISBN: 0136135161
- [14] Sindhgatta, R., Sengupta, B.: An extensible framework for tracing model evolution in SOA solution design. In: OOPSLA Companion, pp. 647–658 (2009)
- [15] Laskey, K.: Considerations for SOA Versioning. In: 2008 12th Enterprise Distributed Object Computing Conference Workshops, September 16, 2008, pp. 333–337. IEEE (2009)
- [16] Dam, H.K., Ghose, A.: Supporting Change Propagation in the Maintenance and Evolution of Service-Oriented Architectures. In: 17th Asia Pacific Software Engineering Conference (APSEC) 2010, November 30-December 3, pp. 156–165. IEEE (2010)
- [17] Hirzalla, M.A., Zisman, A., Cleland-Huang, J.: Using Traceability to Support SOA Impact Analysis. In: 2011 IEEE World Congress on Services (SERVICES), July 4-9, pp. 145–152. IEEE (2011)
- [18] Orriëns, B., Yang, J., Papazoglou, M.P.: Model driven service composition. In: Orłowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 75–90. Springer, Heidelberg (2003)

- [19] Ravichandar, R., Narendra, N.C., Ponnalagu, K., Gangopadhyay, D.: Morpheus: Semantics-based Incremental Change Propagation in SOA-based Solutions. In: IEEE International Conference on Services Computing, SCC 2008, July 7-11, pp. 193–201. IEEE (2008)
- [20] Ramesh, B., Jarke, M.: Toward reference models for requirements traceability. *IEEE Transactions on Software Engineering* 27(1), 58–93 (2001)
- [21] Zimmermann, O., et al.: Managing architectural decision models with dependency relations, integrity constraints, and production rules. *Journal of Systems and Software* 82(8), 1249–1267 (2009)
- [22] Ahmad, A., Pahl, C.: Customisable transformation-driven evolution for service architectures. In: Proceedings of the European Conference on Software Maintenance and Reengineering (CSMR), pp. 373–376. IEEE Computer Society (2011)
- [23] Zalewski, A., Kijas, S., Sokołowska, D.: Capturing Architecture Evolution with Maps of Architectural Decisions 2.0. In: Crnkovic, I., Gruhn, V., Book, M. (eds.) ECSA 2011. LNCS, vol. 6903, pp. 83–96. Springer, Heidelberg (2011)

The *LVA-Index* in Clustering

Piotr Lasek

Institute of Computer Science, University of Rzeszów,
Pigonia 1, 35-310 Rzeszów
lasek@univ.rzeszow.pl

Abstract. In this work we describe the application of the *LVA-Index* in the *NBC* algorithm and discuss the results of the relevant experiments. *LVA-Index* is based on the idea of approximation vectors and the layer approach. *NBC* is considered as an efficient density-based clustering algorithm. The efficiency of *NBC* is strictly dependent on the efficiency of determining nearest neighbors. For this reason, the authors of *NBC* used the simplified implementation of the *VA-File* and the idea of layers for indexing points and determining nearest neighbors. We noticed that it is possible to speed up the clustering by applying the *LVA-Index* which provides the means for determining nearest neighbors faster. The results of the experiments prove that incorporating the *LVA-Index* into the *NBC* improves the efficiency of clustering.

1 Introduction

Clustering is considered one of most important methods in knowledge discovery. It is used in many different areas including data mining, document retrieval, image segmentation and pattern classification [1]. The goal of clustering is to group objects into different sets of similar points according to certain criteria. There are different types of clustering algorithms: hierarchical, partitioning, density-based and grid-based.

Some examples of hierarchical clustering algorithms are: *BIRCH* [2] and *CURE* [3]. The former uses clustering features and a clustering feature tree (*CF-tree*) to represent clusters. It is quite efficient but can only find spherical clusters. The latter achieves better clustering quality. To model a cluster and compute distances between clusters, *CURE* uses so called representative points. By using these points *CURE* is able to discover clusters of any shape. The family of partitioning clustering algorithms can be divided into *k*-means algorithms and *k*-medoids algorithms [1]. In *k*-means algorithms, clusters are represented by the gravity center of the cluster; in *k*-medoid algorithm, clusters are represented by the center objects of the cluster. One of the partitioning clustering algorithms is *CLARANS* [4], which is an improved *k*-medoid algorithm. The computational complexity of *CLARANS* is $O(kN^2)$, which makes it inefficient.

In order to locate clusters, the density-based algorithms use a density function. For example, in the density-based algorithm *DBSCAN* [5], an object must fulfill

the following basic criterion in order to be a member of a cluster : the minimum number of objects k must be located within the neighborhood of this object. *DBSCAN* has several disadvantages: since the neighborhood radius and the minimum number of object within the neighborhood are predetermined, *DBSCAN* cannot distinguish small, close and dense clusters from large and sparse clusters. In other words, the parameters are of a global sense; *DBSCAN* has a low efficiency when working with high-dimensional databases. Nevertheless, the quality of clustering when using *DBSCAN* is quite good. There also exists an extension to *DBSCAN - OPTICS* [6], which computes an augmented cluster ordering for automatic and interactive cluster analysis.

Grid-based algorithms (like *STING* [7]) divide the data space into rectangular cells using a hierarchical structure. The computational complexity of *STING* is $O(N)$. However the quality of clustering is low as the relationship between neighboring cells is not considered.

The *NBC* [8] algorithm belongs to the density-based clustering algorithms and its authors claim that it fulfills, so called, *3-E* criteria, namely:

- 1) effectiveness - it can discover clusters of any shape and discern clusters of different local-densities and multi-granularities in one dataset
- 2) scalability - it can handle large and high-dimensional databases
- 3) ease of use - it needs only one input parameter - the k value

The authors of *NBC* used a cell-based structure and *VA-File* [9] to organize the data, which they say makes it efficient and scalable. However, having analyzed the source code of the *NBC* algorithm [10], we have discovered that the implementation of the internal index allows us to determine neighbors belonging only to the first layer of a given cell. Such a simplification may seem to lead to faster clustering but there may exist datasets (for example with different local densities) where searching for the nearest neighbors in layers with numbers greater than 1 can be needed. However, we can demonstrate such examples (see Section 3) in which the results are better even when using layers farther away from the first layer.

This work is divided into the 5 sections. In Section 2 the *LVA-Index* [11] and the *NBC* algorithm are recalled. Then, in Section 3, a description of how the *LVA-Index* was incorporated into *NBC* is presented. The experiments are described and discussed In Section 4. In Section 5, conclusions are drawn and further works are presented.

2 Application of *LVA-Index* in *NBC*

In this section we shortly recall both *LVA-Index* and the *NBC* algorithm. Then, we give an overview of what changes were needed in order to incorporate our *LVA-Index* into *NBC*.

2.1 The *NBC* Algorithm

The *LVA-Index* is designed so that each non-empty cell representation contains the list of the references to the points P belonging to it and the list of structures representing its l nearest layers ($L_i, i = 1, 2, \dots, n$). Simultaneously, each point p representation stores the reference to a cell, to which p belongs. Such a structure is presented in Figure 1.

In what follows, depending on the context, we use the terms such as *cell*, *layer*, *point*, etc., interchangeably with the *representation of a cell*, *representations of a layer*, *representations point*, etc.

The number n of the nearest layers, for each non-empty cell, is determined experimentally and depends on the number of dimensions d and the density of the dataset, so that the number of points in l neighbor layers is equal to or greater than k , where k is the number of the nearest neighbors to be found.

In our implementation of *LVA-Index*, only non-empty cells are stored in the structure.

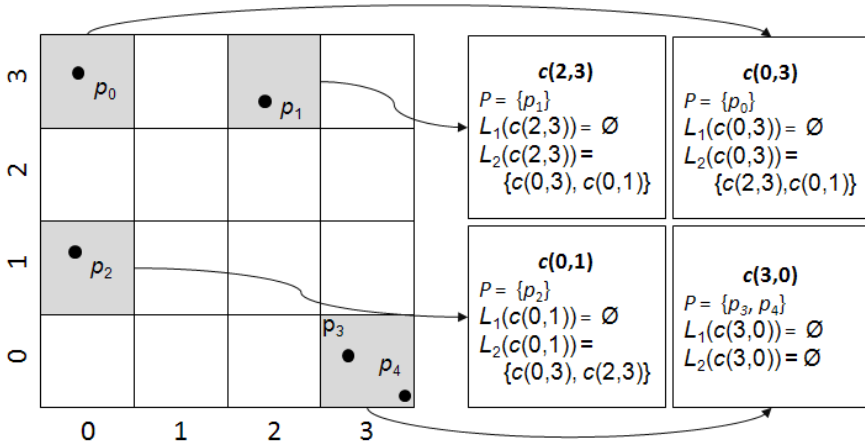


Fig. 1 The structure of *LVA-Index* for the two-dimensional data space ($n = 2, d = 2$)

2.2 The *NBC* Algorithm

The *NBC* belongs to a group of the density-based clustering algorithms and was proposed by Zhou, Zhao, Guan and Huang in [8]. *NBC* begins with calculating the *kNB* (*k-Neighborhood*), *R-kNB* (*Reverse k-Neighborhood*) and *NDF* (*Neighborhood-based Density Factor*) factors for each point from the input dataset (Figure 2). In order to calculate each of those factors the nearest neighbors must be determined. For the sake of efficiency, the authors applied the layer approach, so that the dataset is represented as a set of cells. Each of these cells has neighbor layers according to Definition 2 from [11].

```

NBC(Dataset, k)
for each object p in Dataset p.clst no = NULL;
CalcNDF(Dataset, k); // calculate NDF
NoiseSet.empty(); // initialize the set for storing noise
cluster_count = 0; // set the first cluster number to 0
for each object p in Dataset
{ // scan dataset
  if (p.clst no != NULL or p.ndf < 1) continue;
  p.clst no = cluster_count; // label a new cluster
  DPSet.empty(); // initialize DPSet
  for each object q in  $kNB(p)$ 
  {
    q.clst no = cluster_count;
    if (q.ndf >= 1) DPSet.add(q)
  }
  while (DPSet != empty)
  { // expanding the cluster
    p = DPSet.getFirstObject();
    for each object q in  $kNB(p)$  {
      if (q.clst no != NULL) continue;
      q.clst no = cluster_count;
      if (q.ndf >= 1) DPSet.add(q);
    }
    DPSet.remove(p);
  }
  cluster_count++;
}
for each object p in Dataset
{ // label noise
  if (p.clst no is NULL) NoiseSet.add(p);
}

```

Fig. 2 The NBC algorithm (after [1])

The second part of the algorithm is based on the results of calculating *NDF* factors (Figure 3). In this part the entire dataset must be scanned and according to the values of the *NDF* factors, objects are assigned to appropriate clusters. *NBC* is capable of determining clusters of different densities which distinguishes this algorithm from the others. It is also effective and scalable because as the size of the dataset grows, the time-cost of *NBC* increases at a slower rate than in case of e.g. DBSCAN [5].

```

CalcNDF(Dataset, k)


---


partition(Dataset)
for each object p in Dataset
{
  Map p into an appropriate cell c;
  p.cell = c;
  c.ObjectSet.add(p.id);
}
for each object p in Dataset
{ // evaluate kNB(p)
  CandidateSet.empty();
  layer_no = 0;

  while(CandidateSet.size() < k)
  {
    Add the objects in cells of Layer(p, layer_no) to CandidateSet;
    layer_no++;

    while(CandidateSet != empty)
    {
      Update kNB(p) from CandidateSet;
      CandidateSet.empty();
    }
    for each new object q in kNB(p)
      for each cell c in Layer(q, 1)
        if (objects in c have not been added to CandidateSet before)
          Add the objects in c to CandidateSet;
  }
  for each object q in kNB(p)
    q.SizeOfRkNB++; // update the size of R-kNB(q) }
}
for each object p in Dataset
  NDF(p)=p.SizeOfRkNB / p.SizeOfkNB; // calculate NDF

```

Fig. 3 The *CalcNDF* method of the *NBC* algorithm (after [1])

2.3 Implementation Details

As already mentioned above, following our analysis of the source code and private communication with the authors of *NBC*, the authors' implementation of the *NBC* algorithm used only one layer when searching for the nearest neighbors. For this reason, comparing the original implementation with ours could give wrong results, thus we decided to implement the *NBC* by algorithm by ourselves. We implemented the algorithm as it was presented in Figure 1, and simultaneously, not changing its structure, we identified the sections of the algorithm responsible for determining nearest neighbors and used the *LVA-Index* In those cases.

In Figures 2-3, the places where the *LVA-Index* was used both for searching the nearest neighbors and determining the layers have been underlined. It can be noticed that the influence of the index for the whole clustering algorithm may be very crucial since index methods are invoked from different sections of the algorithm.

Moreover, the calls to the index appear in many nested loops. It is also important to mention that when using the *LVA-Index* with parameter n (the maximum number of layers to be scanned when searching the nearest neighbors) equal to 1, it will be equivalent to the original implementation of *NBC*. In other words, when referring to the reference (original) implementation of *NBC*, we will assume that the *LVA-Index* with parameter n equal to 1 will be used.

3 Experiments

In this section we present the results of experiments that we carried out. Due to the fact that the *LVA-Index* takes several parameters such as: k – the minimum number of objects in a cluster, b - the number of bytes per dimension, and n - the number of nearest neighbors to be found, we have performed a large number of experiments. For this reason, only the most interesting results are presented in this section. Additional results are presented in Appendix A in [10].

We have also performed experiments for other indices, e.g. *VA-File* and *R-Tree*. These experiments were run on the same dataset containing 2658 objects. The times measured and presented in Figure 5 correspond to the clustering results presented in Figure 4. It can be seen that the shortest clustering time was achieved using the *LVA-Index*-based implementation.

In each of the experiments performed which tested the original implementation of *NBC* and its versions (*LVA-Index*, *VA-File*, *R-Tree*), four clusters were found. There were minor differences between clusters concerning specific features and parameters of the indices, such as: the number of bits per dimension (b), the maximum number of layers stored in the index (n), the different index structures (*R-Tree*). Table 1 presents the parameters and runtimes measured. We ran two series of experiments using two-dimensional dataset: in the first series we varied the value of n from 3 to 1; in the second, we varied the value of b from 5 to 7.

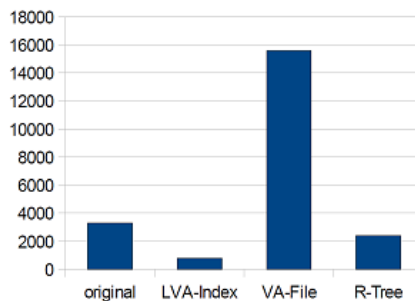


Fig. 4 The times of the clustering for the test dataset for different indices, $k = 20$; for *LVA-Index* $l=3$

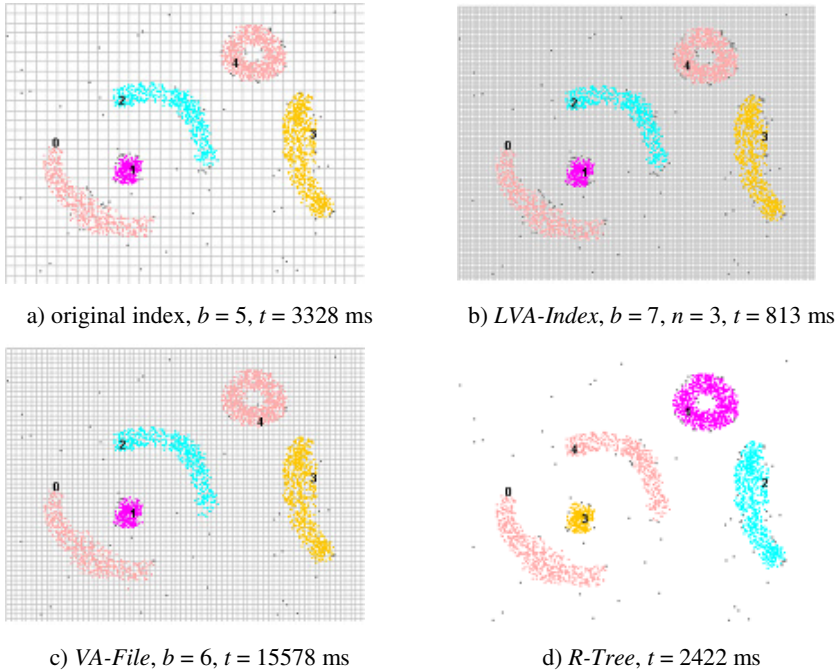


Fig. 5 The times of the clustering for the test dataset for different indices ($k = 20$)

In Figure 6 it can be seen that the good clusters' separation for two dimensional dataset was achieved when the value of b was equal to 7 and the value of n was equal to 3. Such a quality of clustering was not reached when using the reference implementation of *NBC* (6c-6f) with n equal to 1.

In conclusion, we can state that changing the value of n slightly increases the time-cost of the clustering. However, as we have shown, when changing the number of bits per dimension (b), the clustering quality can be improved even without adversely affecting the clustering time-cost.

Table 1 The times of clustering and values of parameters used (k - number of nearest neighbors to be found, b - number of bits per dimension, n - maximum number of layers, t - clustering time given in milliseconds, * - the original *NBC*)

Figure	k	b	n	t
3a	20	7	3	812
3b	20	7	2	515
3c	20	7	1*	235
3d	20	5	1*	2953
3e	20	6	1*	875
3f	20	7	1*	203

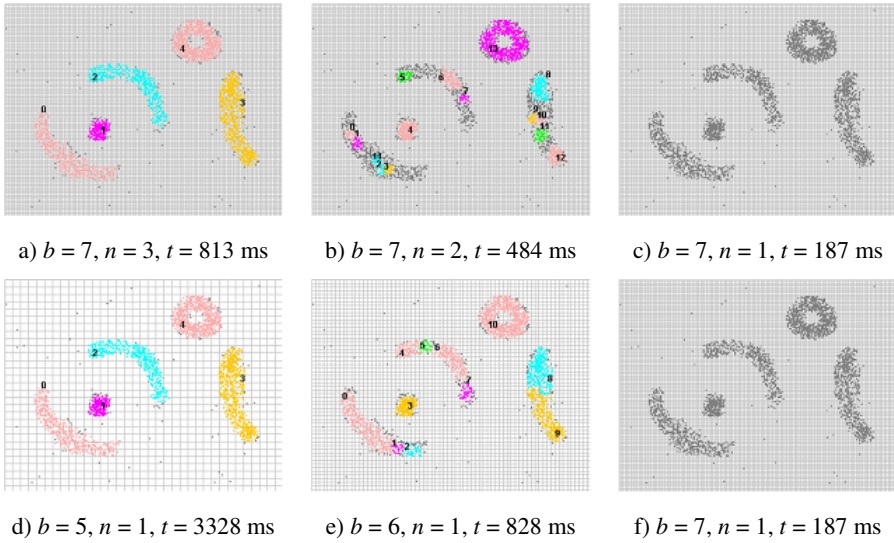


Fig. 6 The results of the clustering for the test dataset ($k=20$)

4 Conclusions and Further Works

Our experiments show that applying *LVA-Index* to *NBC* with appropriate values of parameters, improves the quality and effectiveness of clustering even up to 100 times for low dimensional datasets. This result was achieved by using our iterative approach for enumeration of cells belonging to a given layer in *LVA-Index*.

Our future research will involve focusing more on the influence of the value of n parameter on the quality and effectiveness of clustering. Cases, where the value of n is too small and there is a need for determining layers "on the fly" during clustering could be also examined. Such a case could employ the idea of Triangle Inequality property in a way as described in [12]. Regardless of clustering, *LVA-Index* due to its ability to efficiently determine cells belonging to a given layer, could be applied in the domain of the online analytical processing.

References

- [1] Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann Publishers (2000)
- [2] Zhang, Q., et al.: BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery* 1(2), 141–182 (1997)
- [3] Guha, S., Rastogi, R., Shim, K.: CURE: an efficient clustering algorithm for large databases. In: *SIGMOD 1998: Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pp. 73–84 (1998)

- [4] Ng, R., Han, J.: CLARANS: A Method for Clustering Objects for Spatial Data Mining. In: CLARANS: A Method for Clustering Objects for Spatial Data Mining, pp. 1003–1016 (2002)
- [5] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of International Conference on Knowledge Discovery and Data Mining, pp. 226–231 (1996)
- [6] Ankerst, M., Breunig, M., Kriegel, H., Sander, J.: OPTICS: Ordering Points To Identify the Clustering Structure. In: SIGMOD Conference, pp. 49–60 (1999)
- [7] Wang, W., Yang, J., Muntz, R.R.: STING: A Statistical Information Grid Approach to Spatial Data Mining. In: Proceedings of the 23rd International Conference on Very Large Data Bases, August 25–29, pp. 186–195 (1997)
- [8] Zhou, S., Zhao, Y., Guan, J., Huang, J.: A neighborhood-based clustering algorithm. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 361–371. Springer, Heidelberg (2005)
- [9] Weber, R., Schek, H.J., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: Proceedings of the 24th VLDB Conference on Very Large Data Bases (VLDB 1998), New York City, NY, pp. 194–205 (1998)
- [10] Lasek, P.: Efficient Density-Based Clustering. Ph.D. Thesis, Wydawnictwo Politechniki Warszawskiej (2011)
- [11] Lasek, P.: LVA-Index: an Efficient Way to Determine Nearest Neighbors. In: Man Machine Interactions, ICMMI (2009)
- [12] Kryszkiewicz, M., Lasek, P.: A neighborhood-based clustering by means of the triangle inequality. In: Fyfe, C., Tino, P., Charles, D., Garcia-Osorio, C., Yin, H. (eds.) IDEAL 2010. LNCS, vol. 6283, pp. 284–291. Springer, Heidelberg (2010)

Three Different Approaches in Pedestrian Dynamics Modeling – A Case Study

Robert Lubaś, Janusz Miller, Marcin Mycek, Jakub Porzycki, and Jarosław Wąs

AGH University of Science and Technology,
Institute of Applied Computer Science,
al. Mickiewicza 30, 30-059 Kraków
{rlubas,miller,mycek,porzycki,jarek}@agh.edu.pl

Abstract. In this work different approaches to crowd dynamics modeling are compared in terms of efficiency and accuracy. The authors analyze and test applicability of some characteristic microscopic models including: Generalized Centrifugal Force Model, Social Distances, as well as macroscopic model represented by hydrodynamic approach. Models were compared on a real life test case, for which precise empirical results were obtained, to find sufficient balance between dependability of results and computational effort.

1 Introduction

As the global population grows, capacity of buildings, size of public events and gatherings grows as well. Furthermore rapid development of means of interpersonal communication in the last century [1], especially boom of so-called social networks in the last decade such as most known Facebook, dramatically changed dynamics of public gatherings life cycle. Because of that reaction time available to public services in case of emergency is decreasing. Such situation brings necessity of developing models for crowd safety analysis that are both fast and accurate.

Despite rapid development in this field of knowledge, predictive capability is still too low. On the one hand low fidelity models are fast to compute, but fail to capture small scale events which can be critical to predict in time emergency situations. On the other hand, more complex, high fidelity models manage to capture such phenomena, but are very time-consuming to compute. Hence, choice of most appropriate model is crucial to obtain dependable results in reasonable time.

We can ask the following question: is it possible to create an fast simulation with sufficient fidelity and accuracy?

Three various models characterized by different level of fidelity were chosen for further analysis.

One macroscopic model:

Hydrodynamic Approach – continuous model based on observation that crowd in high densities behaves like a fluid [2],

Two microscopic models:

Generalized Centrifugal Force Model – continuous self-driven multi particle system based on psychological field theory [3].

Social Distances Model – based on non-homogeneous cellular automata approach to pedestrian dynamics making use of proxemics rules [4, 5].

For a test case free evacuation of lecture hall was chosen. During experiment average outflow and evacuation time was measured. Geometry and initial number of people in room were shared as parameters between models, as well as mean free velocity of pedestrians.

2 Proposed Models of Pedestrian Dynamics

2.1 Macroscopic Approach

Macroscopic models of pedestrian movement take inspiration from hydrodynamics or gas-kinetic theory. The state is described by locally averaged quantities - density $\rho = \rho(t, x, y)$, and mean velocity $v = v(t, x, y)$ - regarded as dependent variables of time and space. The density has to satisfy a hyperbolic partial differential equation invoking the mass conservation law. Interactions between pedestrians are represented either by a system of partial differential equations (e.g. Helbing [6] or Bellomo and Dogbé [7]) or by closure relations for the average velocity of individuals in terms of the density and its gradient (e.g. Hughes [8] or Coscia and Canavesio [2]).

The model described in this work is similar to the one presented by Coscia and Canavesio [2]. It is a macroscopic, first-order model of crowd dynamics in bounded domain for two-dimensional flow-problem. It takes into account two fundamental aspects of pedestrian movement. On the one hand, pedestrians aim toward specific target, which determines the main direction of motion v_0 but on the other, they tend to avoid crowding (this deviation is represented by a vector v_1). The primary direction of motion is determined by the shortest way to the target. A pedestrian tends to maintain a preferential direction of motion toward target he wants to reach, but at the same time he is disposed to slightly deviate from it in order to avoid crowding. Coscia and Canavesio [4] take this into account including a minimum directional derivative in the visual range of the pedestrian. In contrast to this approach we suggest representing a circumvent of crowding by a vector

$$v_1 = (\bar{\rho} - 1)\nabla\bar{\rho}, \quad (1)$$

where $\bar{\rho}$ is a nondimensionalized ρ with respect to ρ_{\max} . To sum up, the preferred direction of motion of the individuals u is specified by the weighted sum

$$u = v_0 + \alpha v_1, \quad (2)$$

where α is a parameter of the model (we assume $\alpha = 0.8$ [m²/s]).

A magnitude of the pedestrians' velocity is a scalar function of density. To describe this dependency we assume the Kladek function [9]

$$\varphi(\rho) = v_{des} \cdot \left[1 - \exp\left(-\gamma\left(\frac{1}{\rho} - \frac{1}{\rho_{max}}\right)\right)\right], \tag{3}$$

where v_{des} is the free (desired) speed of a pedestrian and γ denotes a parameter of the model. Like in [9] we assume $\gamma = 1.913 \text{ [m}^2\text{]}$.

Finally the motion of pedestrians in presented model is described by two equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0, \tag{4}$$

which expresses the principle of conservation of mass (pedestrian), and

$$v = \frac{\varphi(\rho)}{\|u\|} u, \tag{5}$$

which links the velocity to the local density conditions.

Partial differential equation (4) is supplemented by boundary conditions of Dirichlet type for exit gate and Neumann type in case of walls or another obstacles (in presence of obstacles, they are understood as internal boundaries to the walking area).

2.2 Centrifugal Force Model

Social Force Models. Social Force models of pedestrian movement, first introduced by Helbing et al [10] are based on simple analogy to Newton's laws of motion. First law states that when an object experiences no *net force*, then it is either at rest or it moves in a straight line with constant speed. By this analogy, if pedestrian changes his velocity in the presence of other pedestrians, one can model this interaction as a social interaction force. In mathematical terms, the change of velocity v_i in time t is given by the acceleration equation:

$$m_i \frac{dv_i}{dt} = F_i^D + \sum_{j \neq i} F_{ij}^I + \sum_W F_{iW}^I, \tag{6}$$

where m_i denotes mass of a pedestrian, $F_i^D = m \frac{v_i^0 - v_i}{\tau_i}$ is a driving term respon-

sible for keeping desired velocity v_i^0 with some time constant τ_i . F_{ij}^I is an interaction term induced by other pedestrian and finally F_{iW}^I describing interaction between pedestrian and surrounding walls W .

Centrifugal Force Model. Original Social Force model [10] assumed exponential with regard to relative distance interaction term. Yu, Chen, Dong and Dai [11] proposed different interaction term based on the fact that with dimension analysis only one dimensionless quantity can be constructed for acceleration, speed and relative distance:

$$F_{ij}^I = -m_i K_{ij} \frac{V_{ij}^2}{\|R_{ij}\|}, \quad (7)$$

where V_{ij}^2 is relative velocity of pedestrians, $\|R_{ij}\|$ is distance between pedestrians and K_{ij} is a coefficient which takes into account field of vision of pedestrians:

$$K_{ij} = \frac{1}{2} \left[\frac{V_i \cdot e_{ij} + \|V_i \cdot e_{ij}\|}{\|V_i\|} \right], \quad (8)$$

where e_{ij} is a versor pointing from one pedestrian to another:

$$e_{ij} = \frac{R_{ij}}{\|R_{ij}\|}, \quad (9)$$

Name of the model comes from familiarity of this interaction term to centrifugal force known from classical physics. This approach was adopted and developed by Chraïbi et al. [3] in Generalized Centrifugal Force Model which is used in this work, where in addition pedestrian shape depends on their velocity.

2.3 Social Distances Model

In the models based on cellular automata, space is divided into square cells and one cell can be occupied by only one pedestrian [12]. Pedestrian movement is mainly determined by the current configuration of the neighborhood. Thanks to that, this model is extremely efficient. Most of cellular automata models are based on square lattice based on 40 cm cells. Fidelity of the classical model is higher than in macroscopic models, but space representation is relatively coarse.

To improve fidelity of classical CA models, Social Distances model was proposed [4]. In the model each pedestrian is represented as ellipse placed in square lattice. The model was then adapted for modeling the evacuation of large objects [12, 13].

In the presented model pedestrians are represented as a part of multi-agent systems using some rules of asynchronous and non-homogeneous cellular automaton. Each pedestrian (agent) has its own independent attributes: speed, direction, destination, distance traveled, and evacuation time. The model described in this section is: *discrete, microscopic, numerical and stochastic*.

Due to the fact that the model uses the theory of cellular automata is unavoidable discretization of space. It is impossible to accurately represent real world on a two-dimensional discrete square grid of size 25 cm.

Because of potential gradient layer, is possible to choose the next cell and pedestrian can make a move. Each cell, on which the pedestrian can move, has a value increasing proportionally to the distance from the exit.

In the model agent is represented by an ellipse, whose center is situated in the middle of a single cell. Pedestrian may move to another cell in the Moore neighborhood of radius 1, taking into account the field of vision, which is wide at 180° , and is defined for different positions $0^\circ, 45^\circ, 90^\circ, 135^\circ, 180^\circ, 225^\circ, 270^\circ, 315^\circ$.

The crucial issue is to establish the set of forbidden and allowed positions for all cells in Moore neighborhood of radius 1, each cell being occupied by one person [4]. The calculation of the allowed/forbidden positions is based upon simple geometrical dependencies [13]. It takes into account the following: the orientations of two ellipses occupying two adjacent cells and the size of their cross-section. It is assumed that the position is allowed if the ratio of the calculated cross-section (for this position) to the size of the ellipse is smaller than imposed tolerance $\mathcal{E}_N \in [0,1]$.

3 Experimental Data and Simulation Results

Models mentioned above were compared with a real life test case. In 240 seats lecture hall an experimental evacuation of approximately 210 students was conducted. Experiment was carried out under normal conditions as *announced drill* (according to [15]).

Fig. 1 shows simplified plan of the lecture hall used for the experiment. There are 15 rows, with 16 seats each. Grey areas indicate place unavailable for pedestrians. During the experiment only one half of the lower exit was opened, while upper doors were closed.

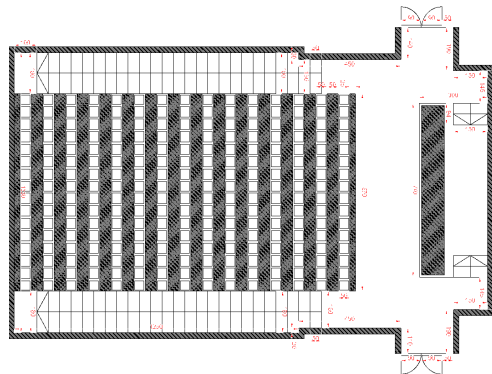


Fig. 1 Lecture hall plan with dimensions

Total evacuation time was 162 seconds. Fig. 2 represents observed outflow¹. The highest observed peak slightly exceed 2.5 persons per second. However, on average it is approximately 1.35 persons. Trend line shows that at the beginning outflow reach average level of 1.7, but then as the crowd density increases, outflow decreased to approximately 1.2.

Another important observed feature are significant oscillations of outflow with average amplitude of ~1.3 and period of 8-10 sec. This phenomenon is caused by forming waves of denser and sparser crowd.

The outflow data from experiment was compared with analogical outflow data obtained from simulations of: macroscopic model Fig. 3, Centrifugal Model Fig. 4 and Social Distances Model Fig. 5. Highest similarity of the flow function to real data was observed in Social Distances Model and slightly lower similarity in Centrifugal Model. In macroscopic model similarity of flow function was lowest.

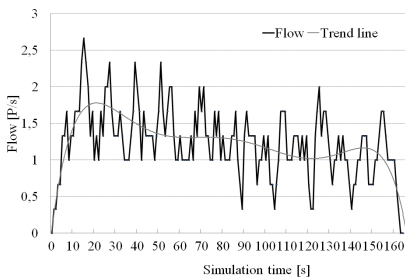


Fig. 2 Real data - statistics of pedestrians outflow measured in experiment

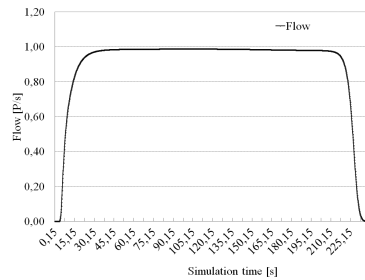


Fig. 3 Macroscopic model - statistics of pedestrians outflow

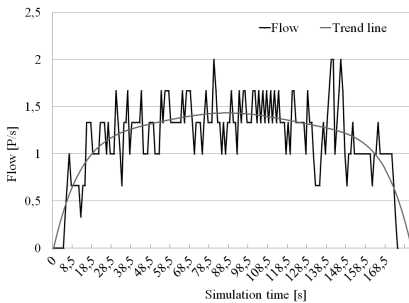


Fig. 4 Centrifugal model - statistics of pedestrians outflow

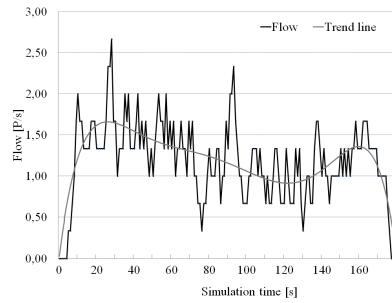


Fig. 5 Social Distance Model - statistics of pedestrians outflow

¹ To reduce high frequency noises, value for given second N is calculated as an average of data for seconds N-1, N and N+1.

Analysis of trend lines² shows overall tendencies of outflow changes. For experimental data it varies from 1.7 to 1.2 decreasing as crowd density increase. Similar phenomenon could be observed in Social Distance Model. Although in this method outflow is in range 2.05 - 1.6, one can observe outflow decreasing in time. Other methods, Centrifugal Force Model and macroscopic approach, don't show this phenomena.

Table 1 Total and partial evacuation times for all simulated models and real life experiment. Total evacuation time is measured from beginning of the experiment to the moment when there is nobody in the room. Partial evacuation times for stairs are defined similarly - it is time after which there is no one on the stairs. Model parameter V_{des} describes pedestrians desired velocity.

Model	V_{des} [m/s]	Total evacuation time	Left stars time	Right stairs time
Macroscopic Model	0.98	230	87	199
	1.11	198	73	172
	1.34	171	61	149
Social Distances	0.9	172	80	160
	1.11	131	66	120
	1.34	105	58	103
Centrifugal	0.9	173	108	164
	1.11	160	94	154
	1.34	158	98	146
Experimental		163	83	129

Total evacuation time for every used model is close to empirical data. Comparison of evacuation times in particular models with experimental data are presented in Table 1. Parameter V_{des} describes value of pedestrians desired velocity, total evacuation time is time measured from start of evacuation to the last person passing through the exit, and respectively left and right stairs time - flow time on stairs measured from the first person who appears on the stairs, to the last person who leaves the stairs.

One can notice that the different methods have different sensitivity to the change of the parameter of desired velocity V_{des} . Macroscopic model and Centrifugal model are less sensitive than Social Distances model.

Fig. 6 illustrates evacuation times gained from different models. Centrifugal and Social Distances Model have similar statistics, while macroscopic model has slightly different results.

² Trend line was calculated as 5th order polynomial approximation.

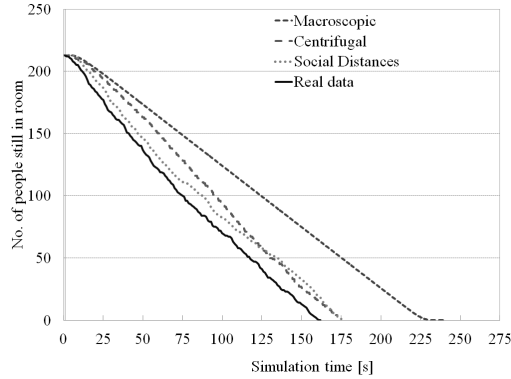


Fig. 6 Evacuation times - comparison of all models for $V_{des} = 0.9$ m/s and real data (experimental results)

Considering the performance of each method, it should be noted that the most effective is macroscopic model and its demand for computational power is constant during an executed simulation (Fig. 7).

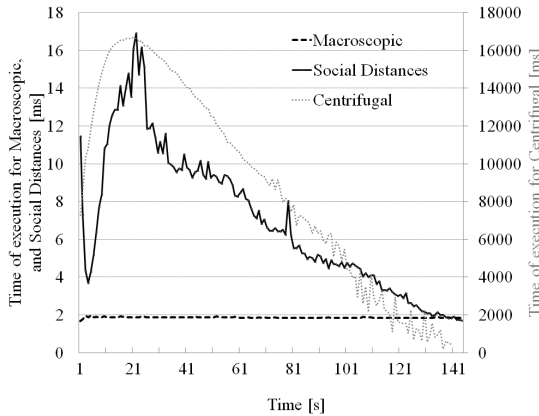


Fig. 7 Performance tests for all models. Plot shows execution time for every consecutive second of simulation. For macroscopic and Social Distances models axis of performance is placed on the left side of chart, while for Centrifugal model axis of performance is placed on the right side of the graph.

Although demand for computing power for Social Distances and Centrifugal differs significantly in scale, both of them have similar characteristic points Fig. 7. The maximum is associated with *the highest number of interactions and collisions among pedestrians* in both *microscopic models*, as the largest contribution to the computational complexity.

Centrifugal Force Model has lowest computational efficiency from all models as can be seen on Fig. 7 Social Distances and macroscopic models have similar efficiencies - two orders of magnitude lower.

4 Concluding Remarks

Three completely different approaches in modeling of evacuation are presented in the article. The first one; macroscopic approach based on hydrodynamics - crowd is represented as a fluid with its motion characterized by using differential equations. The second approach is Generalized Centrifugal Model based on microscopic Social Force Model, when all pedestrians are represented as moving, interacting particles. The third approach is Social Distances model based on non-homogeneous cellular automata where pedestrians are represented as ellipses placed on a square lattice.

The most accurate representation of space is achieved in Generalized Centrifugal Model, next in Social Distances model and the least accurate is macroscopic model. This means that fidelity of the models is changing from highest level in Molecular Dynamics based Centrifugal Model, through middle level - Cellular Automata model, to macroscopic level. It should be stressed, that presented case study it is not optimal size of the simulation environment because macroscopic model is more often used for vast environments.

In terms of performance, the most efficient method is macroscopic model - based on the principles of hydrodynamics, whilst somewhat lower performance presents Social Distances model (but it is the same order of magnitude as macroscopic model). Among the models tested, the least efficient model is Centrifugal Force Model and it is a cost of the most precise representation of pedestrians positions and velocities.

Thus, it should be noted, that the accuracy/fidelity of a model is closely related to its performance. The more accurate is the model; the lower is its performance. Comparisons of models plays important role in choosing best one that guarantee highest dependability of produced crowds simulation systems both in terms of efficiency and fidelity.

Important issue is to clarify the difference in the results of evacuation times received using various methods, as well as potential sources of errors. An important role is played by parameterization of each model, obtained as a result of the calibration process. Each model obviously requires further, more precise calibration to adapt to the specific conditions.

Acknowledgement. This work is partially supported under the FP7 ICT program of the European Commission under grant agreement No 231288 (SOCIONICAL).

References

- [1] Gwizdalla, T.: *International Journal of Modern Physics C* 17(12), 1791–1799 (2006)
- [2] Coscia, V., Canavesio, C.: First order macroscopic modeling of human crowd dynamics. *Math. Models Methods Appl. Sci.* 18, 1217–1247 (2008)
- [3] Chraïbi, M., Seyfried, A., Schadschneider, A.: Generalized centrifugal-force model for pedestrian dynamics. *Phys. Rev. E* 82, 046111 (2010)
- [4] Waś, J., Gudowski, B., Matuszyk, P.J.: Social distances model of pedestrian dynamics. In: El Yacoubi, S., Chopard, B., Bandini, S. (eds.) *ACRI 2006. LNCS*, vol. 4173, pp. 492–501. Springer, Heidelberg (2006)
- [5] Waś, J., Lubaś, R., Myśliwiec, W.: Proxemics in discrete simulation of evacuation. In: Sirakoulis, G.C., Bandini, S. (eds.) *ACRI 2012. LNCS*, vol. 7495, pp. 768–775. Springer, Heidelberg (2012)
- [6] Helbing, D.: A fluid-dynamic model for the movement of pedestrians. *Complex Systems* 6, 931–415 (1992)
- [7] Bellomo, N.: On the modeling crowd dynamics from scaling to hyperbolic macroscopic models. *Math. Models Methods Appl. Sci.* 18(suppl.), 1317–1345 (2008)
- [8] Hughes, R.L.: The flow of large crowds of pedestrians. *Mathematics and Computer in Simulations* 53, 367–370 (2000)
- [9] Weidmann, U.: *Transporttechnik der Fussgänger Transporttechnische Eigenschaften des Fussgängerverkehrs, Literaturauswertung, Schriftenreihe des IVT. Technical Report 90*, Institut für Verkehrsplanung und Transportsysteme, Zürich (1992)
- [10] Helbing, D., Farkas, I., Vicsek, T.: Simulationg dynamical features of escape panic. *Nature* 407, 487 (2000)
- [11] Yu, W.J., Chen, R., Dong, L.Y., Dai, S.Q.: Centrifugal force model for pedestrian dynamics. *Phys Rev. E* 72, 026112 (2005)
- [12] Schadschneider, A., Klingsch, W., Kluepfel, H., Kretz, T., Rogsch, C., Seyfried, A.: *Evacuation Dynamics: Empirical Results, Modeling and Applications*. In: *Encyclopedia of Complexity and Systems Science*. Springer Science+Business Media, New York (2009)
- [13] Waś, J., Lubaś, R., Myśliwiec, W.: Towards realistic modeling of crowd compressibility. In: Peacock, R.D., Kuligowski, E.D., Averill, J.D. (eds.) *Pedestrian ad Evacuation Dynamics*. Springer US (2011)
- [14] Burstedde, C., Klauch, K., Schadschneider, A., Zittartz, J.: Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications* 295(3–4), 507–525 (2001)
- [15] *SFPE, SFPE handbook of fire protection engineering*. The Society of Fire Protection Engineers (2008)

The End-To-End Rate Adaptation Application for Real-Time Video Monitoring

P. Lubkowski and Dariusz Laskowski

Military University of Technology,
Warsaw, 00-908 Warsaw, Gen. S. Kaliskiego 2 Street,
{Piotr.Lubkowski,Dariusz.Laskowski}@wat.edu.pl

Abstract. In modern advanced monitoring solutions, video data are shared across IP networks. Such systems are used for connection with remote offices or other locations and provide information from multiple video sensors. However, a transmission of data from multiple cameras may lead to the degradation of video quality and even to lack of transmission abilities especially in heterogeneous networks. This paper proposes a concept of application with end-to-end rate adaptation for video monitoring systems ensuring accessibility and retainability of video service even in the limited capacity of transmission system.

1 Introduction

Video surveillance belongs to the realm of security and relies on the process of monitoring the behavior of people, objects and processes in the selected area. A characteristic feature of the video monitoring systems is their performance on an ongoing basis. A relatively brief interruption of the surveillance system may determine the overall effectiveness of the solution. Requirements for video surveillance systems are defined in the area of functional and quality constraints. Quality assurance at the network level (QoS – Quality of Service) is extremely important because it allows prediction of the size of the available bandwidth and the level of losses at the application layer. However, this information will only allow a fair distribution of bandwidth between the cameras and does not reflect the user's requirements on service severability. Thus in the case of video surveillance systems it is important to consider also the quality at the application layer. The QoS at the application layer is defined in ITU-T Recommendation G.1010 and includes parameters related to the establishing, disconnecting and call-blocking of connection. Within this group of parameters, service accessibility and service retainability are particularly relevant. They determine the possibility of obtaining on demand services in the specified range and under certain conditions, and to continue it for the required period. This approach allows optimization of the quality from the end user perspective (QoE – Quality of Experience). From the perspective of QoE, there are three ways of ensuring the quality.

The first method assumes the alignment of the video data rate on the basis of scaling rate. The second method assumes the stream protection against losses by addition of redundant information. And the last method involves the monitoring of packet loss at the application level and selective retransmission of selected parts of the video stream. This article is limited only to the problem of bandwidth scalability because of the presented issues complexity.

Scaling the video bit rate can be implemented in three domains: 1) compression – by increasing the quantization parameter QP, 2) time – changing the number of frames per second, 3) spatial – by reducing the resolution. The use of compression was the subject of the following publications [2], [6], [8]. An example of application of H.264/AVC video codec implementation and the impact of compression on the video quality is shown in [12]. The effect of change of resolution on the QoE quality is presented in [7]. The change of the frame rate was analyzed in many works, because it is one of the most important parameters to the faithful reproduction of the movement. In [14], a model is presented in which the number of frames per second and the amount of information transmitted for each frame are changed.

These methods involve optimization of the quality of a single video stream under the assumption that it is provided a link of a specified quality. Unfortunately, the channel capacity of video surveillance systems may vary during transmission. Thus, the channel capacity should be taken into account in the process of scaling the video bit rate for quality optimization. It can be solved by using cross-layer design to update application layer with channel capacity information. The use of cross-layer design for optimization of video streaming is broadly discussed in many papers concerning especially wireless networks. In [1] CLO (Cross Layer Optimizer) is proposed that collects information concerning resources available at MAC layer and optimizes video parameters according to them. Another solution called QoS-Enabled Controller is given in [3]. The video quality is optimized according to the parameters taken from LINK (MAC) and PHY (radio) layers. In our proposal we decided to use our own cross-layer mechanism discussed in [11]. It uses a Cross-Layer Data Base (CLDB) and cross-layer signaling (CLS) to share information's between LINK and APP layers.

In this paper, a End-to-end Rate Adaptation (EREA) application is presented which is characterized by using cross-layer cooperation between network and application layers for preserving service accessibility and retainability.

2 The Concept of Application for Rate Adaptation

As it was already mentioned, the concept of application is based on cross-layer signaling between layers and implements a series of procedures that result in the adaptation of video bit rate to the available throughput of the transmission channel. The EREA application consists of two main modules called server and client, respectively. The server module retrieves the image from the supplied IP camera. Based on information provided by the procedure of channel utilization, it

performs the adaptation of the video stream and then sends a video to the client module. The client module supports a connection to the server module in order to obtain real-time video from the supported IP cameras. The client module performs also a link testing procedure. The effect of its actions is the information about the accessible bandwidth, which is then available to the server. Communication between server and client as well as between the server and IP cameras is implemented using the RTSP protocol. The general functional diagram of EREA application is shown in Fig. 1.

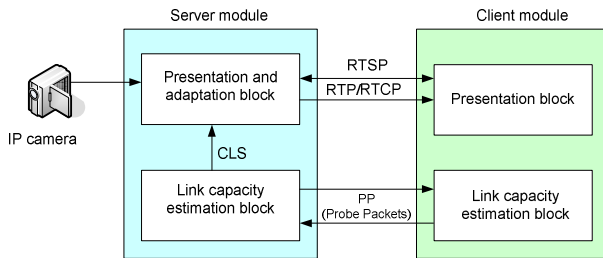


Fig. 1 Functional diagram of EREA application

Server module realizes procedures of video adaptation and channel estimation. These procedures are performed in a block of video adaptation and the block of link capacity estimation, respectively. The video adaptation block accepts an invitation to a session from the client side. Bandwidth estimation procedure is started automatically just after the call with the client is being established. It involves sending a series of test packets of known size and duration toward the client module.

The client module consists of a video sequences presentation block and bandwidth estimation block. The video sequences presentation block displays the stream of data obtained from the server module. The link capacity estimation block receives a series of probe packets (PP) from the server module in the predefined period of time. These packets are sent using TCP connection and determined by the size of the package and its duration. It should be noted that connection is identified by the source and destination IP address and number of used ports. The time of arrival of the first and last packet in the series are written on the client module.

Based on the time difference and the amount of data transmitted in the series, it is possible to calculate the accessible bandwidth. The calculated bandwidth is then sent by TCP connection to the server module. There, this value is passed over cross-layer signaling to the adaptation block which performs resampling of video frames on the base of bilinear interpolation.

3 Link Capacity Estimation Procedure

In this paragraph, the focus is set on end-to-end capacity estimation between server and client modules. This estimation is supported by the server with the aid of additional signaling between the entities. Measurements are performed only if a new real-time video stream should be started and they are initiated on demand of the client and evaluated using probe packets.

The motivation for this approach is the assumption of guaranteeing the continuity of data transmission from video monitoring. The capacity estimation may support clients in situations when there will be a temporary lack of available bandwidth or the transmission will be realized by so-called bottle-neck links. The techniques for evaluation of bandwidth capacity are well known and widely discussed. Good examples can be found in [13] where VPS (Variable Packet Size) probing, PPTD (Packet Pair/ Train Dispersion) probing and TOPP (Trains of Packet Probing) are presented. Packet Pair Probing technique is also used for the estimation of bandwidth experiment described in [10]. A particular modification of its was given in [9]. We use simplified version of this technique in our EREA application. As mentioned before, testing procedure uses a series of probe packets to evaluate a link capacity toward a client. Diagram of the throughput estimation evaluation is shown in Fig. 2.

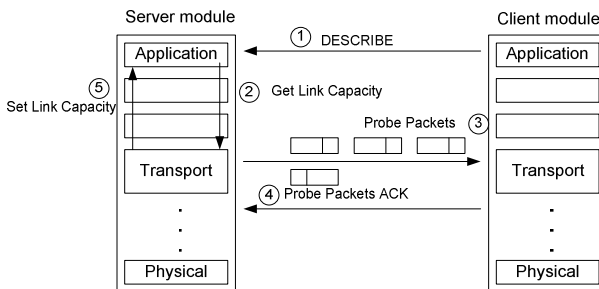


Fig. 2 Diagram of throughput estimation

The procedure of throughput estimation is triggered as a new client starts a RTSP session with server (1 – DESCRIBE). This results in sending a signal to start a testing procedure which is performed in the transport layer (2). Next, a series of 10 PP is sent toward a client module (3). Each packet has the size of 1KB and is sent every 500ms. The average available throughput T is evaluated in the client module based on the information on the number of received data and the time difference between the received packets. The resulting throughput is sent back in the TCP connection to the server module (4) and passed to a block of video rate adaptation (5). The procedure for throughput testing is started every 0.5 seconds, so the server module has always the actual knowledge about the available bandwidth. The total size of the transmitted test data reduces the bandwidth for video transmission by about 20kb/s.

4 Video Rate Adaptation Procedure

The procedure of video rate adaptation to the available throughput is one of the most important and most complicated procedures implemented in the EREA application. It utilizes a bilinear interpolation mechanism. Each pixel of the resulting image has a value calculated as the averaged value of the four neighboring points of the input image. In fact, this mechanism performs a vertical and horizontal linear interpolation. As a result, the video frame is modified in the area of resolution. Due to lossy compression, the speed of video stream is adjusted to the actual throughput. This mechanism allows a satisfactory quality with relatively small computational complexity.

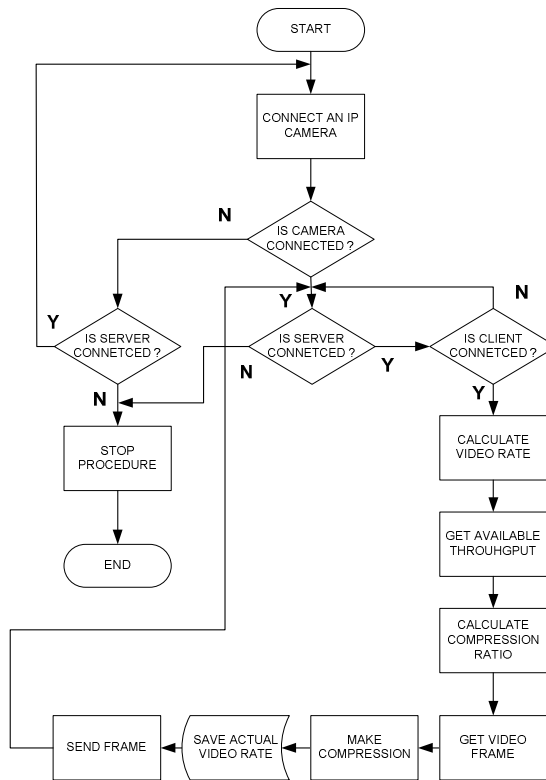


Fig. 3 Block diagram of video rate adaptation procedure

The block diagram of the video rate adaptation procedure is presented in Fig. 3. At the beginning, the procedure checks if the IP camera, server and client are connected. After that, an actual video rate is calculated as well as an available throughput is taken from the procedure of link capacity estimation. In the next step, a calculation of compression ratio is carried out. The calculated value is subsequently used in the compression of the video frames. At the end of this process, the actual video rate is stored and frame is send toward a client module.

The modules of server and client mentioned above were implemented in the form of executable programs written in C++ and configured to operate under Windows. Running the application requires the installation of additional C/C++ libraries included in the “*libjpeg-turbo-1.2.0-vc*” application. This application accelerates the video encoding and decoding as well as floating point operations, so it is possible to use the full power of x86 computing.

5 Validation and Verification of EREA Application

All components of the EREA application were implemented in the demonstrator established in the MUT (Military University of Technology) laboratory within the INSIGMA project. The main goal of the project is the development and implementation of a complex information system for comprehensive detection and identification of risks and the monitoring and identification of moving objects. The demonstrator consists of three access domains, which are connected by a backbone network. The overall architecture of demonstrator is presented in Fig. 4.

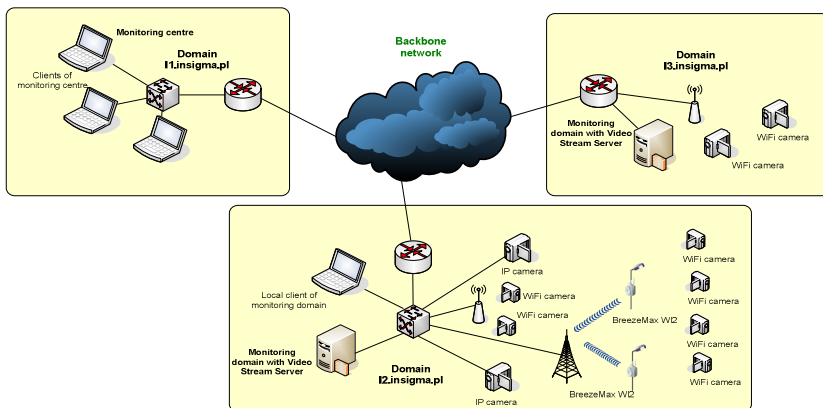


Fig. 4 The overall architecture of demonstrator

Two domains are a source of information from video monitoring and one of them represents a monitoring centre. The domains are connected over backbone network where tunnels of predefined bandwidth are established. Only under this condition we can use our mechanism for available bandwidth evaluation.

Fig. 5 shows the simplified structure of a measuring stand. It contains two PCs that are representing the server and client as well as an IP camera representing the source of video information. The camera continuously monitors the selected area and provides the image in the form of real-time video to the server.

A set of experiments and scenarios proposed for verification and validation of the developed EREA application were conducted. The tests were divided into two groups: 1) subjective assessment of the image quality, 2) objective estimation of the end-to-end QoS and QoE. Detailed description of each group of tests is given in Table 1.

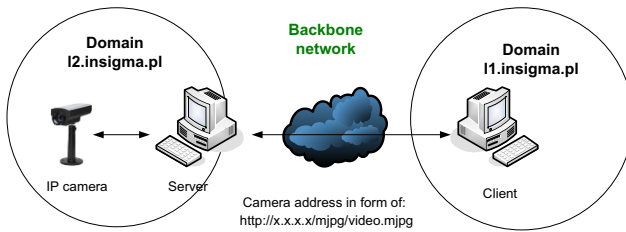


Fig. 5 Simplified structure of a measuring stand

Table 1 Practical experiment scenarios

#	Experiment	Description
1)	Subjective assessment of the image quality	Evaluation of image quality with MoS (Mean Opinion Score)
2)	Objective assessment of the end-to-end QoS and QoE	Measurement of end-to-end delay between client and server
		Assessment of image quality with PSNR (Peak Signal-to-Noise Ratio)

The image is derived from an Axis Q1755 IP camera. The initial resolution is set to 1280x720. The server performs additional video compression using bilinear interpolation based on the designated bandwidth. Throughput restriction is performed by the client application. It reduces the bandwidth of the client interface to the value declared in the application window. The measurement results are related to quality indicators defined in the following recommendations [4], [5].

The first experiment selected for qualitative discussion in this paper is based on a video stream transmitted to client application with different quality. It represents the image quality changes caused by adaptation of video stream bit rate according to throughput measured in the transmission channel. The quality of reference image is presented in Fig. 6. The bit rate of video stream is equal to 1200 kb/s.

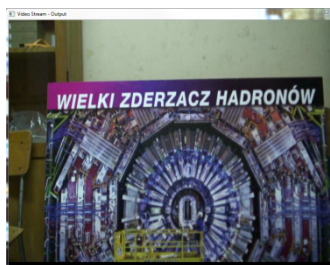


Fig. 6 Reference image shown in the server application window

Fig. 7 presents the image shown by the client application with 50% and 90% reduction of quality, respectively. It can be seen that in the case of 50% quality reduction there is no significant degradation of image quality. A deterioration of the image is invisible, we can observe a high focus and good contrast. Therefore, from the MOS perspective, this means score 5. In this case, the measured bit rate is about 500 kb/s. However, for 90% reduction, visible artifacts can be observed with respect to the reference image (the fields marked with a red border). The bit rate of the video stream is reduced to 120 kb/s. This time the rating is equal to 3.

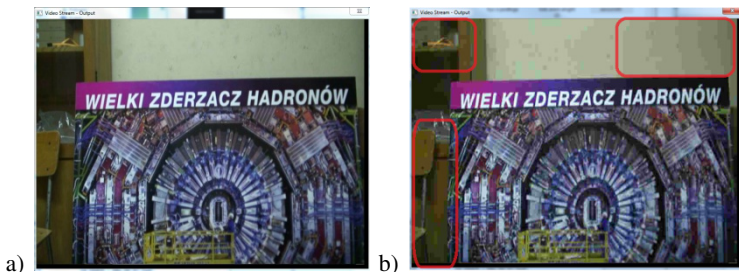


Fig. 7 Image quality shown in client application window (a – 50% quality reduction, b – 90% quality reduction)

Qualitative tests of EREA application incorporate also the measurement of end-to-end delay of a video packet. The next figure shows the delay as a function of throughput changes (Fig. 8). As expected, the delay value increases with the decrease in throughput. It can be seen that rapid growth of delay occurs at the throughput below 400 kb/s. This results from the increase in waiting time for probe packet transmission and the need of video frames buffering. A picks of delay observed at this graph are also caused by the disruption in the transmission of PP packets. The measured values of delay remain in the acceptable range.

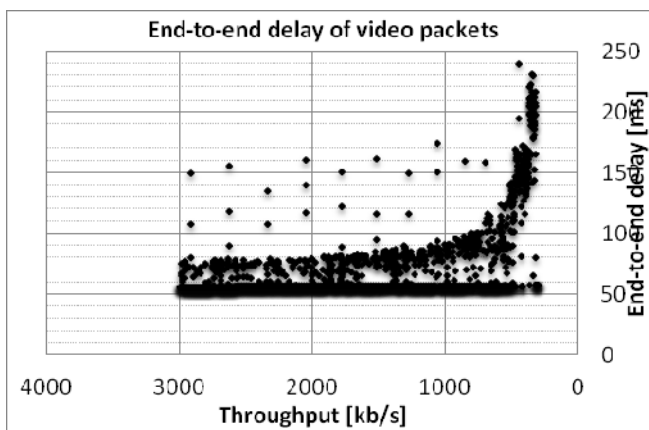


Fig. 8 End-to-end delay of video packets

The next figures show the result of the assessment of image quality with PSNR. This parameter is used to determine the level of similarity between the reference image and the compressed one, and is expressed in decibels (dB). In practice, the smaller value of coefficient means lower similarity. For the measurements of PSNR, a MSU Video Quality Measurement Tool 3.0 application was used. Fig. 9 shows the value of PSNR for the 90% and 50% image quality. Increasingly bright colors can be noticed at 50% image quality, which means a greater difference of images.

An even greater lack of fit can be seen at 2% of image quality. However, the value of PSNR is still high, indicating a high probability of correct identification of the information contained within the image.

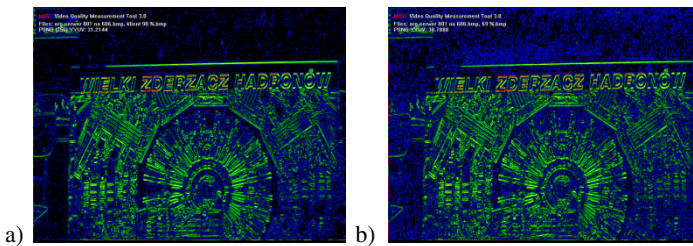


Fig. 9 Result of the comparison of the images with MSU application (a – PSNR = 31.21, b – PSNR = 30,06)

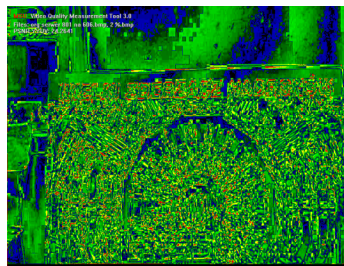


Fig. 10 Image representing PSNR in the quality of 2% (PSNR = 24.26)

6 References

Testing results presented here confirm that the elaborated EREA application work correctly in conditions of changing the throughput of the transmission channel. From these results we can see that video monitoring can be ensured using proposed video frame adaptation and throughput measurements.

The results obtained show also that the additional compression of video stream preserves the continuity of service without significant deterioration of image quality. This enables the support of end-to-end QoS in terms of service accessibility and retainability. The proposed application is part of QoS supporting

architecture for INSIGMA network which in connection with resource manager and admission control modules will protect the quality of video monitoring transfers. The application will be further developed in the direction of handling multiple video streams and recording them. It is worth mentioning that the application will also enable the adaptation of video bit rate on demand of the client.

Acknowledgement. The work has been supported by the European Regional Development Fund within INSIGMA project no. POIG.01.01.02.00.062/09.

References

- [1] Abdallah, A., et al.: Cross layer Design for Optimized Video Streaming Over Heterogeneous Networks. In: IWCMC 2010 Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, pp. 933–938 (2010)
- [2] Leontaris, A., Reibman, A.R.: Comparison of Blocking and Blurring Metrics for Video Compression. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, March 18–23, vol. 2, pp. 585–588 (2005)
- [3] Hsu, C., Hefeeda, M.: Cross-layer optimization of video streaming in single-hop wireless networks. In: Proceedings of the ACM/SPIE Multimedia Computing and Networking, MMCN 2009 (2009)
- [4] ITU-T Recommendation G.1010, End-User Multimedia QoS Categories Series G: Transmission Systems and Media, Digital Systems and Networks Quality of Service and Performance-Study (2001)
- [5] ITU-T Recommendation P.800, Methods for subjective determination of transmission quality (1996)
- [6] Pandel, J.: Measuring of flickering artifacts in predictive coded video sequences. In: Ninth International Workshop on Image Analysis for Multimedia Interactive Services, pp. 231–234. IEEE Computer Society, Washington, DC (2008)
- [7] Janowski, L., Romaniak, P., Papir, Z.: Content Driven QoE Assessment for Video Frame Rate and Frame Resolution Reduction, Multimedia Tools and Applications. Springer (2011)
- [8] Farias, M.C.Q., Mitra, S.K.: No-reference video quality metric based on artifact measurements. In: IEEE International Conference on Image Processing, ICIP 2005, vol. 3, pp. III-141–III-144 (September 2005)
- [9] Lies, M., Sevenich, P., Barz, C., Pilz, M., Łubkowski, P., Bryś, R., Milewski, J.: The Effective QoS Mechanism For Real Time Services in IP Military Network. In: RCMCIS (2004)
- [10] Hu, N., Steenkiste, P.: Estimating available bandwidth using packet pair probing (2002), <http://citeseer.ist.psu.edu/hu02estimating.html>
- [11] Łubkowski, P., Krygier, J., Amanowicz, M.: Provision of QoS for multimedia services in IEEE 802.11 Wireless network. In: Information System Technology Panel Symposium on "Dynamic Communications Management" IST-062, Budapest (2006)
- [12] Romaniak, P., Janowski, L., Leszczuk, M., Papir, Z.: Perceptual Quality Assessment for H.264/AVC Compression. In: CCNC 2012 FMN, 4th International Workshop on Future Multimedia Networking, Las Vegas, NV, USA, January 14 (2012)

- [13] Prasad, R.S., Murray, M., Dovrolis, C., Claffy, K.: Bandwidth estimation: metrics, measurement techniques, and tools. *IEEE Network* 17, 27–35 (2003)
- [14] Inazumi, Y., Yoshida, T., Sakai, Y., Yuukou, H.: Estimation of the optimal frame rate for video communications under bit-rate constraints. *Electronics and Communications in Japan (Part I: Communications)* 86(12), 54–67 (2003)

Discrete Transportation Systems Quality Performance Analysis by Critical States Detection

Jacek Mazurkiewicz and Tomasz Walkowiak

Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
{Jacek.Mazurkiewicz,Tomasz.Walkowiak}@pwr.wroc.pl

Abstract. The paper presents the analysis of discrete transportation systems (*DTS*) performance. The formal model of the transportation system is presented. It takes into consideration functional and reliability aspects. Monte Carlo simulation is used for estimating the system quality metric. The quality of the system is assessed in three levels, as: operational, critical and failed. The proposed solution allows to predict the system quality within the short time horizon. The paper includes numerical results for real mail distribution system.

1 Introduction

The paper analysis discrete transportation systems, i.e. a transportation system in which goods are transported by vehicles of limited capacity. The vehicles carry commodities between destinations according to some management rules [9].

The commodities are fixed in size (the volume of commodities is always a discrete number) and they appear in the system in random time moments. The main aim of the system is to deliver goods within given time limit. [9][11][12]

We focus here on the problem how to predict the system quality within the short time horizon. The quality of the system is measured by its performance, i.e. a ratio of on-time deliveries [11]. The quality performance (for example defined by a service level agreement) is achieved when the ratio of on-time delivers is not smaller than a given threshold. Therefore, we could define two states of the system: operational (the system is fulfilling the quality requirements) and failed (requirements are not fulfilled). Since we are predicting the future, we could only measure the probability of being in a given state.

Therefore, we propose to extend this two state approach by the third (critical) state which corresponds to the situation when the system is “being at a turning point, or a sudden change” [8]. In other word critical state is connected with the transition between qualitatively different states [3] (operational and failed).

The critical states could be used as an indicator for the system management that the system urgently needs maintenance action [1].

The paper is organized as follows. The section two presents the *DTS* model and the brief description of the each part of the system. Next we focused on the quality analysis – done by computer simulation which calculates the performance metric. Then, we present the discussion about the critical states of the operation based on actual conditions pointed for the state. The theory is verified by the practical case study – the real transportation system dedicated to post services located at Lower Silesia region in Poland.

2 DTS Model

A realization of the transportation system service needs a defined set of technical resources. Moreover, the operating of vehicles transporting commodities between system nodes is done according to some rules – some management system. Therefore, we can model discrete transportation system as a 4-tuple [7]:

$$DTS = \langle Client, Driver, TI, MS \rangle \quad (1)$$

where: *Client* – client model, *Driver* – driver model, *TI* – technical infrastructure, *MS* – management system.

Technical infrastructure includes set of nodes with defined distances between them and a set of vehicles. Each vehicle is described by its load (number of containers) and random parameters which model vehicle breakdowns (requiring repair by one of the maintenance teams) and traffic congestion (which result in random delays in the transportation time). The service realized by the clients of the transport system is mail sending from some source node to some destination one. Client model consists of a set of clients. Each client is allocated in the one of nodes creating the transportation system [10]. The client allocated in an ordinary node generates containers (a given amount of commodities, measured in discrete numbers) according to the Poisson process with destination address set to ordinary nodes. In the central node, there is a set of clients, one for each ordinary node. Each client generates containers by a separate Poisson process and is described by intensity of container generation.

The human infrastructure is composed by the set of drivers. So the description of this part of system infrastructure requires the analysis of the drivers' state and the algorithms, which model the rules of their work. Each driver could be in one of following states (s_d): rest (not at work), unavailable (illness, vacation, etc.), available (at work – ready to start driving), break (during driving), driving. The number of driver working hours is limited by the labour law. The daily limit for each driver equals to 8 hours and a single driver operates with one truck. Drivers work in two shifts, morning or afternoon one.

Moreover we propose to categorise the driver's illnesses as follows: short sick: 1 to 3 days, typical illness: 7 to 10 days, long-term illness: 10 to 300 days [6]. We prepare the daily record of the driver. The decisions (send a truck to a given destination node) are taken in moments when a container arrives to the central node.

The truck is sent to a trip if: the number of containers waiting in for delivery in the central node of the same destination address as that just arrived is larger than a given number, there is at least one available vehicle, the simulated time is between 6 am and 22 pm minus the average time of going to and returning from the destination node.

The truck is sent to a node defined by destination address of just arrived container. If there is more than one vehicle available in the central node, the vehicle with size that fits the best to the number of available containers is selected, i.e. the largest vehicle that could be fully loaded. If there are several trucks with the same capacity available the selection is done randomly. The restriction for the time of truck scheduling (the last point in the above algorithm) are set to model the fact that drivers are working on two 8 hours shifts.

3 DTS Quality Analysis

The analysis of DTS is done by computer simulation. It allows to observe different parameters of the analyzed system and calculate metrics which could be used to assess the system quality.

3.1 DTS Simulation

The simulation algorithm is based on tracking of all system elements. The state of each vehicle could be one of the following: waiting for a task to be realized, approaching the ordinary or the central node, waiting to be unloaded and loaded at the ordinary or at the central node, waiting for a repair crew, being repaired. The state of each node is a queue of containers waiting to be delivered. The state is a base for a definition of an event, which is understood as a triple: time of being happened, object identifier and state. Based on each event and states of all system elements rules for making a new event has been encoded in the simulation program. The random number generator was used to deal with random events, i.e. failures or vehicle journey time.

It is worth to notice that the current analyzed event not only generates a new event but also could change time of some future events (i.e. time of approaching the node is changed when failure happens before). The event-simulation program could be written in general purpose programming language (like C++), in fast prototyping environment (like Matlab) or special purpose discrete-event simulation kernels.

One of such kernels, is the Scalable Simulation Framework (*SSF*) which is a used for *SSFNet* [11] computer network simulator. *SSF* is an object-oriented API - a collection of class interfaces with prototype implementations. It is available in C++ and Java. For the purpose of simulating *DTS* we have used Parallel Real-time Immersive modeling Environment (*PRIME*) [11] implementation of *SSF* due to much better documentation then available for original *SSF*.

Due to a presence of randomness in the *DTS* model the analysis of it has to be done based on Monte-Carlo approach [5]. What requires a large number of repeated simulation. The *SSF* is not a Monte-Carlo framework but by simple re-execution of the same code (of course we have to start from different values of random number seed) the statistical analysis of system behavior could be realized.

3.2 *Quality Performance Metric*

The quality of the system is assessed by its ability to transport commodities on time. Due to assumed grain of model system we observe the containers (a given amount of commodities) during simulation. The container is assumed to be transported on time when a time measured from the moment when the container was introduced to the system to the moment when the container was transferred to the destination is smaller than a guaranteed time of delivery.

To measure the quality of the whole system we propose to use ratio of on-time deliveries in 24 hour time slots. Let $N_d(t)$ denotes the number of containers delivered in the period the day t , and $N_{pd}(t)$ denotes the number of delivered containers on time within the same 24 hours. Therefore, the system quality could be measured by a ratio of on-time deliveries, defined as:

$$a_t = \frac{N_{pd}(t)}{N_d(t) + 1} \quad (2)$$

The denominator includes +1 modification to prevent the ratio go to infinity in case of a full stoppage of the system (i.e. no containers delivered in the analyzed period).

4 *Critical States of Operation*

The word “critical” is linked to the term of “crisis” which refers to a “change of state”, “a turning point” [2] or “being at a turning point, or a sudden change” [8]. It seems the universal definition of the term is not easy and maybe is not possible. The most proper and as close as possible approach to the description of the system situation is based on the systems attributes and weighted combination of reliability and functional features of it. For the discrete transport systems (*DTS*) discussed in the paper the three quality states of the system are defined: operational, failed or critical. This is our new approach to the problem of the critical state description – completely different to our previous works [6].

The aim of this paper is to show a method that will allow to predict the system behavior within a short time (few days) horizon knowing the actual condition described by the results of multi-criteria analysis focused on the owner point of view. It means the goal is to foresee the most sensible direction of the system management or maintenance decisions or operations.

4.1 System Functional State

The functional state of the system S_t at the end of each day t is given by a 3-dimensional vector that includes: the number of drivers nd_t , that are not sick, the number of vehicles nv_t that are operational and number of stored containers in the warehouses nc_t .

$$S_t = [nv_t, nd_t, nc_t]. \quad (3)$$

4.2 Quality States

As it was mentioned in the introduction we propose to analyze the system its functional state to one of three quality states: operational, critical and failed. It is done based on assessing the performance metric defined in (2). Moreover, we planned to use this assessment to predict the system behavior within a short time horizon Δt (few days). Therefore, we propose to assess a functional state as operational one when the probability that a system will fulfill the performance requirements (i.e. the acceptance ratio will be larger than required level α on $t+\Delta t$ day) is larger or equal to a given threshold level θ .

In a similar way, the system state is assumed to be failed if the probability that a system will not fulfill the performance requirements within a given time horizon is larger or equal to a threshold θ . All other states are assumed to be critical one. Let's introduce a more formal definition. For two thresholds $\alpha, \theta \in (0.5, 1)$, a given functional state S_t of system at day t is named as:

$$\begin{array}{ll} \text{operational} & \text{if } P(a_{t+\Delta t} > \alpha) \geq \theta \\ \text{failed} & \text{if } P(a_{t+\Delta t} \leq \alpha) \geq \theta \cdot \\ \text{critical} & \text{otherwise} \end{array} \quad (4)$$

In other words, after noticing that $P(a_{t+\Delta t} \leq \alpha) = 1 - P(a_{t+\Delta t} > \alpha)$, we can define the critical state as a state for which:

$$1 - \theta < P(a_{t+\Delta t} > \alpha) < \theta \quad (5)$$

5 Test Case Analysis

We propose for the case study analysis an exemplar *DTS* based on the Polish Post regional centre in Wroclaw. We have modeled a system consisting of one central node (Wroclaw regional centre) and twenty two other nodes - cities where there are local post distribution points in Dolny Slask Province.

The length of roads were set according to real road distances between cities used in the analyzed case study. The intensity of generation of containers for all destinations were set to 4.16 per hour in each direction giving in average

4400 containers to be transported each day. The vehicles speed was modeled by Gaussian distribution with 50 km/h of mean value and 5 km/h of standard deviation. The average loading time was equal to 5 minutes. There were two types of vehicles: with capacity of 10 and 15 containers. The MTF of each vehicle was set to 20000. The average repair time was set to 5h (Gaussian distribution). We also have tried to model the drivers availability parameters. We have fulfilled this challenge by using the following probability of a given type of sickness - short sick: 0.003, typical illness: 0.001, long-term illness: 0.00025. The tests were realized for the acceptance ratio level $\alpha = 0.95$, $\Delta t = 2$ days and threshold level $\theta = 0.8$. Moreover the number of drivers that are not sick are taken from the set $nd_t \in (85, 120)$, the number of vehicles that are operational is spanned $nv_t \in (45, 60)$ and number of stored containers in the warehouses $nc_t \in (4000, 7000)$ (Fig. 1).

5.1 Results

The results presented in Fig. 1 could be used as an indicator for management decision. If at the end of given day the system is in a state (defined by a number of operational trucks, working drivers and number containers stored in central and ordinary points) that is assigned in Fig. 1 to a critical group (marked as dot) it should raise an alarm for the management.

Occurrence of such state indicates that the probability that within a few days the performance quality will drops below thresholds raises. The system manager could react by increasing the system resources (drivers and/or trucks). The quantitative analysis of such reaction was presented by authors in [6].

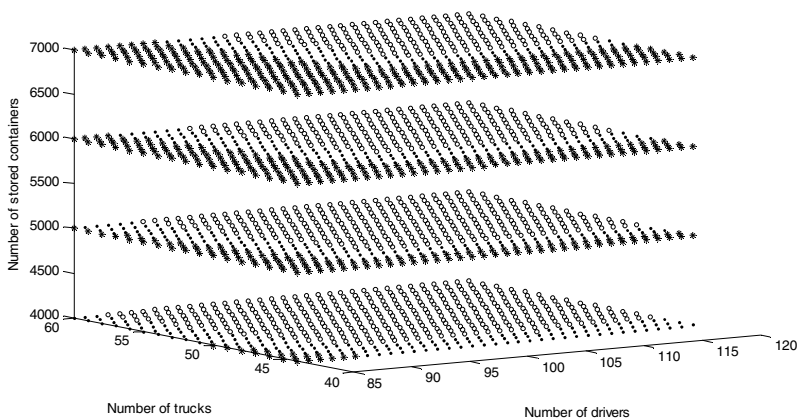


Fig. 1 Assignment of states (triple: number of trucks, drivers and stored containers) to operational (circles), critical (dots) and failed (asterisks) group

6 Conclusions

We have presented a formal model of discrete transportation system (*DTS*) including reliability, functional parameters as well as the human factor component. The *DTS* model is based on Polish Post regional transportation system and reflects all key elements of it with the set of the most important functional and reliability features of them. The critical situation is pointed and described at the necessary level of details by the quality performance parameter. The proposed metric is the source to point the critical states of the system.

The realized analysis based on the real data, i.e. the Polish Post transportation system at Wroclaw area, allowed to predict the system quality within the short time horizon.

The proposed approach allows to perform more deeper reliability and functional analysis of the *DTS*, for example:

- to determine what will cause a "local" change in the system,
- to make experiments in case of increasing volume of goods per day incoming to system,
- to identify weak point of the system by comparing few its configuration,
- to better understand how the system behaves in ordinary and critical situations.

The solution presented can be used as practical tool for defining an organization of vehicle maintenance and transportation system logistics.

Acknowledgement. The presented work was funded by the Polish National Science Centre under grant no. N N509 496238.

References

- [1] Aven, T., Jensen, U.: Stochastic Models in Reliability. Springer, New York (1999)
- [2] Barlow, R., Proschan, F.: Mathematical Theory of Reliability. Society for Industrial and Applied Mathematics, Philadelphia (1996)
- [3] Bouchon, S.: The Vulnerability of interdependent Critical Infrastructures Systems: Epistemological and Conceptual State-of-the-Art. Institute for the Protection and Security of the Citizen, Joint Research Centre, European Commission (2006)
- [4] Burt, C.N., Caccetta, L.: Match Factor for Heterogeneous Truck and Loader Fleets. International Journal of Mining, Reclamation and Environment 21, 262–270 (2007)
- [5] Fishman, G.: Monte Carlo: Concepts, Algorithms, and Applications. Springer (1996)
- [6] Mazurkiewicz, J., Walkowiak, T.: Analysis of Critical Situation Sets in Discrete Transport Systems. In: Kabashkin, I.V., Yatskiv, I.V. (eds.) 12th International Conference: Reliability and Statistics in Transportation and Communication RelStat 2012, Transport and Telecommunication Institute, Riga, Latvia, October 17-20, pp. 354–361 (2012)
- [7] Michalska, K., Mazurkiewicz, J.: Functional and Dependability Approach to Transport Services Using Modelling Language. In: Jędrzejowicz, P., Nguyen, N.T., Hoang, K. (eds.) ICCCI 2011, Part II. LNCS (LNAD), vol. 6923, pp. 180–190. Springer, Heidelberg (2011)

- [8] Stepney, S.: Critical Critical Systems. In: Abdallah, A.E., Ryan, P.Y.A., Schneider, S. (eds.) FASEC 2002. LNCS, vol. 2629, pp. 62–70. Springer, Heidelberg (2003)
- [9] Walkowiak, T., Mazurkiewicz, J.: Algorithmic Approach to Vehicle Dispatching in Discrete Transport Systems. In: Sugier, J., et al. (eds.) Technical Approach to Dependability, pp. 173–188. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław (2010)
- [10] Walkowiak, T., Mazurkiewicz, J.: Analysis of Critical Situations in Discrete Transport Systems. In: Proceedings of International Conference on Dependability of Computer Systems, Brunow, Poland, June 30-July 2, pp. 364–371. IEEE Computer Society Press, Los Alamitos (2009)
- [11] Walkowiak, T., Mazurkiewicz, J.: Functional Availability Analysis of Discrete Transport System Simulated by SSF Tool. *International Journal of Critical Computer-Based Systems* 1(1-3), 255–266 (2010)
- [12] Walkowiak, T., Mazurkiewicz, J.: Soft Computing Approach to Discrete Transport System Management. In: Rutkowski, L., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2010, Part II. LNCS, vol. 6114, pp. 675–682. Springer, Heidelberg (2010)

An Expanded Concept of the Borrowed Time as a Mean of Increasing the Average Speed Isotropy on Regular Grids

Marcin Mycek

AGH University of Science and Technology, Institute of Applied Computer Science,
al. Mickiewicza 30, 30-059 Kraków
mycek@agh.edu.pl

Abstract. In this work it is investigated how the proper choice of the underlying grid for computer simulations in complex systems can increase its observed isotropy; thus increasing dependability of the obtained results. The square lattice with both the von Neumann and the Moore neighborhood as well as the hexagonal lattice are being considered. The average speed isotropy is examined with regard to different length scales. The concept of the borrowed time is reintroduced and expanded for the square lattice with the Moore neighborhood. It is shown that such treatment can decrease the anisotropy of the average speed without increasing complexity of the calculations.

1 Introduction

Discrete representation of space and time in computer simulations holds strong advantage, namely simplicity. It allows faster and larger-scale simulations as compared to the continuous approach [1]. The more complex is the system, the higher is the gain. However, this procedure also introduces errors due to discretization.

When space is discretized, there are three causes of such errors: loss of representation precision when moving from continuous to discrete variables, finite number of directions and arbitrary choice of grid orientation. For this reasons usually a regular lattice is used as it preserves the highest rotational and translational invariance [2]. Still a choice of a proper grid in simulation is not trivial as because of the last two error causes it can induce highly unwanted macroscopic phenomena. This is especially true when the studied system is purely deterministic [3, 4]. Introducing randomness of any kind - be it random distribution of active grid cells [5], a randomized space discretization [6], introducing random walk [7], using inherent randomness such as in lattice-gas automata [8] or asynchronous dynamics [4] can highly increase isotropy. Apart from stochastic solutions decreasing grid cell size can increase both angular and spatial resolution; thus reducing impact of discretization [9]. However, it also significantly increases simulation

time. In this work impact of grid type choice on isotropy of the system will be investigated and modification of the update rules that increase isotropy without increasing computational complexity of the calculations will be proposed.

It should be noted that in this work only square and hexagonal grids are considered, because of their ease of implementation in practical applications. However, triangular lattice is valid for following considerations and will be addressed in future research.

2 The Average Speed Isotropy

2.1 Assumptions and Definitions

It will be assumed that movement on the lattice is restricted to one cell per one time step. For simplicity all used quantities will be unitless. In addition, shortest distance between centers of two adjacent cells and length of one time step are taken to be one. While the reasoning behind this article is universal for any kind of objects moving on the grid - be it gas particles, cars or pedestrians - throughout the text word "agents" will be used as a universal term for such entities.

For movement of agents between two grid cells we can define **an optimal path**:

Definition 1. For a path P_{AB} between two given cells A and B, if there does not exist path connecting those two cells with a shorter time then P_{AB} , it is called an optimal path and denoted by P_{AB}^{op} with its time labeled t_{AB}^{op} .

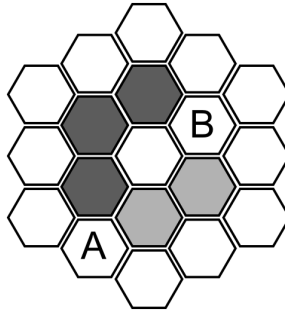


Fig. 1 For cells A and B on hexagonal grid there are three optimal paths (see Def. 1) of length 3. One of them is marked on the figure (light grey) as well as one non-optimal path (dark grey) with length 4.

This is illustrated in **Fig. 1**. Now **average speed** \bar{V}_{AB} for two cells can be defined as:

Definition 2

$$\bar{V}_{AB} = \frac{dist(AB)}{t_{AB}^{op}}, \tag{1}$$

Where $dist(AB)$ means the Euclidean distance between centers of cells A and B. In the limit of infinitesimally small grid cells the average speed between any two cells would be 1. With this fact in mind, relative quantity - **the average speed deviation** D_{AB} can be defined:

Definition 3

$$D_{AB} = \frac{\bar{V}_{AB} - 1}{1}, \tag{2}$$

As not the value of this quantity is of our interest, but its isotropy, some set over which it can be measured have to be defined:

Definition 4. For a given cell A and distance h, a set of all cells having at least one common point with a circle with radius h and center in the middle of the cell A is called **horizon** and denoted by $H_A(h)$.

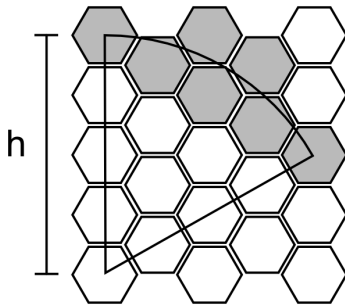


Fig. 2 Horizon $H(4)$ for hexagonal lattice. Only part of the set is shown.

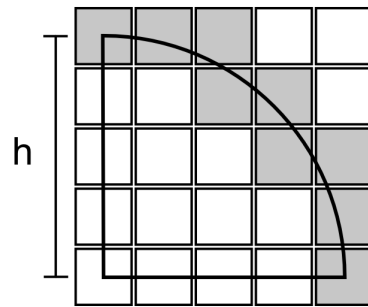


Fig. 3 Horizon $H(4)$ for square lattice. Only part of the set is shown.

The horizon for hexagonal and square lattice is shown in Fig. 2 and Fig. 3 respectively. Then **the measure of the average speed isotropy** - $I_A(h)$ can be finally introduced:

Definition 5

$$I_A(h) = \frac{1}{\#H_A(h)} \sum_{C \in H_A(h)} D_{AC}^2. \tag{3}$$

It is worth noting that such measure is not continuous as it depends on cardinality of the horizon H , which is discrete by definition. If the lattice is large enough, i.e. boundaries have no impact, the measure of the average speed isotropy and the average speed isotropy is independent of the central grid cell choice. Therefore, in the rest of the article it will be assumed that all values are calculated for such grids and cell index will be omitted.

2.2 Hexagonal Grid

A regular lattice with largest natural neighborhood (see Fig. 4) is hexagonal one. For such grid horizon for any given radius is straightforward to calculate. An angular dependence of the average speed deviation can be then determined. Such relation for horizon radii 6.5 and 25 is shown in Fig. 6. and Fig. 7. As can be seen the shape of this relation does not depend on the horizon radius. It is intuitive; as if we take a cell from any horizon multiple of path leading to that cell gives same average speed deviation while belonging to other horizons with larger radius. With this in mind the measure of the average speed isotropy can be calculated for different horizons. Results for horizon radii can be seen in Fig. 8. First of all for horizon radius 1 isotropy is 0 as a horizon contains only natural neighbors of the cell. While an isotropy is decreasing, I quickly obtains a value of about 0.01 for radius 5 and oscillates around it, having almost constant value for larger radii.

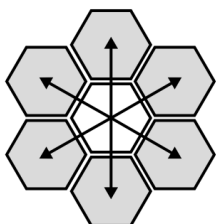


Fig. 4 An example cell on a hexagonal grid and its natural neighborhood

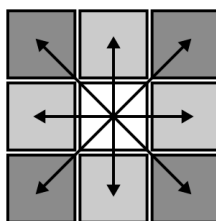


Fig. 5 An example cell on a square grid with natural (von Neumann) neighborhood (light grey) and the extended (Moore) neighborhood (light and dark grey)

2.3 Square Grid

Next two considered grid types are square grids with the natural (von Neumann) and extended (Moore) neighborhoods (see Fig. 5). Repeating a procedure from the last paragraph, one can calculate the angular isotropy for both types of grids (see Fig. 9 and Fig. 10 respectively). For angles close to 45° square grid with the natural neighborhood shows smaller average speed deviation. However, for values of angle lower then 39° it is square grid with the extended neighborhood that shows smaller average speed deviation in terms of an absolute value. Once again a

relation between the measure of average speed isotropy and horizon radius can be calculated. Results for both types of square grids are shown in Fig. 8. As expected from angular relation, the square grid with extended neighborhood shows higher isotropy for non-small horizon radii where a contribution to an isotropy from cells around 45° direction is less dominant. It should be noted that in both cases isotropy is still at least three times lower than for the hexagonal grid for all values of a horizon radius.

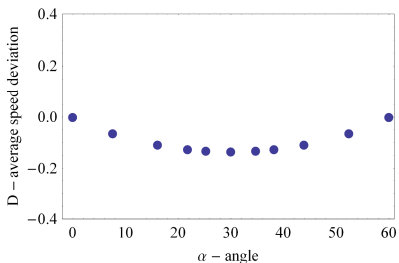


Fig. 6 A relation between the average speed deviation and angle on the hexagonal grid for a horizon H(6.5), where 0° is taken to be a direction straight up

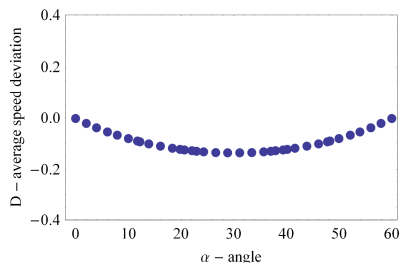


Fig. 7 A relation between the average speed deviation and angle on the hexagonal grid for a horizon H(25), where 0° is taken to be a direction straight up

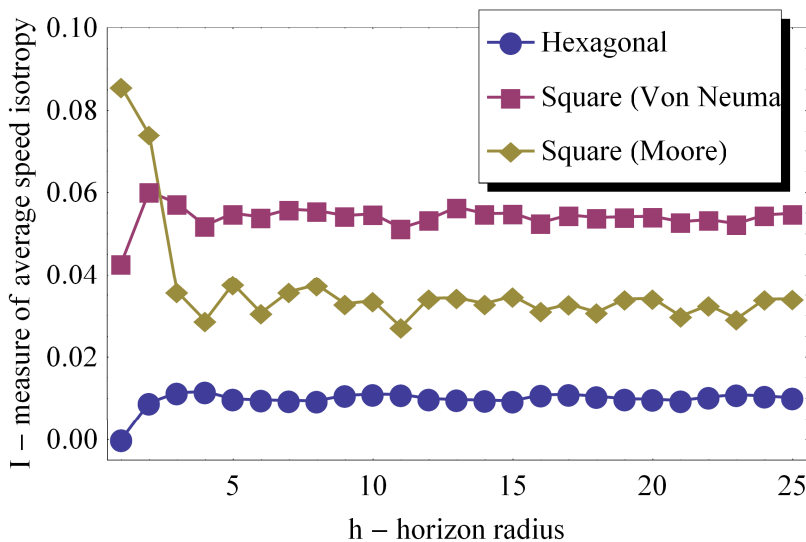


Fig. 8 A relation between the measure of average speed isotropy and a horizon radius for all three grid types - hexagonal, square with the von Neumann neighborhood and square with the Moore neighborhood

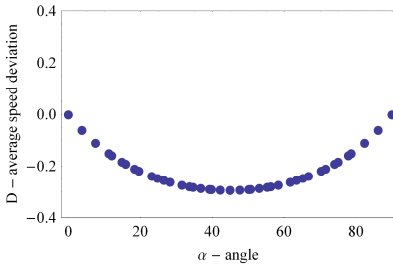


Fig. 9 A relation between the average speed deviation and angle on square grid with the von Neumann neighborhood for a horizon H(15), where 0° is taken to be a direction straight up

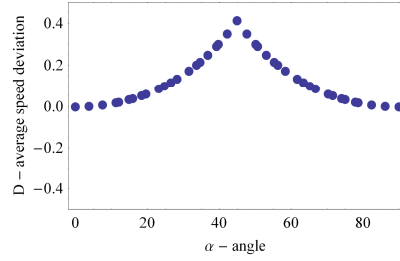


Fig. 10 A relation between the average speed deviation and angle on square grid with the Moore neighborhood for a horizon H(15), where 0° is taken to be a direction straight up

3 Ways to Improve Isotropy

3.1 Concept of the Borrowed Time

To increase isotropy for a square lattice with the extended neighborhood a concept of the "borrowed time" can be introduced [10]. The reasoning is following - it is assumed that speed with which agents move the on lattice is constant. Under this assumption time needed for an agent to move diagonally is longer then one time step. Such move can be allowed if agent is let to "borrow" time to finish the diagonal move. How much time an agent has already borrowed is kept track of. If at some moment this amount exceeds whole time step, an agent is omitted from the next update and his debt is reduced by one time step. Such procedure approximates usage of non-integer time steps. To maintain constant speed the borrowed time parameter b (amount of time an agent has to borrow for a diagonal step) has to satisfy the following condition:

$$\frac{\sqrt{2}}{1+b} = 1. \tag{4}$$

$b = \sqrt{2} - 1$ satisfy this relation. After such update rules modification angular dependency of the average speed deviation for the square lattice can be recalculated taking into account corrected path times. This relation can be seen in Fig. 11. In Fig. 12 the measure of the average speed isotropy with regard to the horizon radius is shown. Isotropy is clearly higher compared to the square lattice with Moore neighborhood without modification and it's even lower then for the hexagonal grid. However, the deviation of the average speed deviation has always negative sign meaning that we get systematical error during calculations.

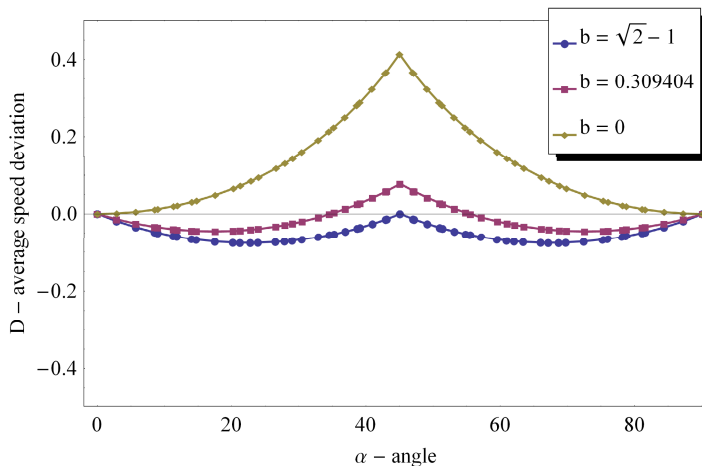


Fig. 11 A relation between the average speed deviation and angle for three different borrowed time parameter values. $b=0$ corresponds to the square lattice with Moore neighborhood without any update rules modification, $b = \sqrt{2} - 1$ is value for the classical borrowed time concept, third value of the borrowed time parameter is one maximizing the average speed isotropy at infinite horizon. 0° is taken to be direction straight up.

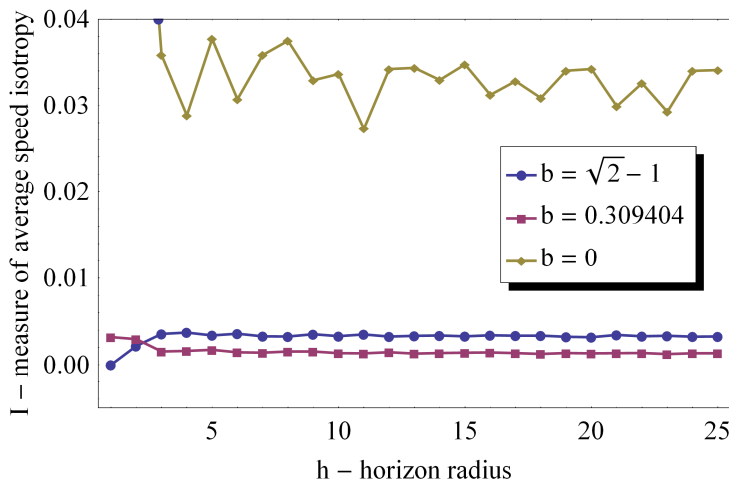


Fig. 12 A relation between the measure of the average speed isotropy and the horizon radius for the square grid with modified update rules for the same three borrowed time parameter values as for Fig. 11

3.2 *Dependence of the Optimal Borrowed Time Parameter on Horizon*

In considered modification of update rules the borrowed time parameter - b was chosen to maintain uniform speed during one time step. However, if agents during whole simulation move more then once, average quantities are of our interest not temporary ones. One can ask what should be the borrowed time parameter to achieve maximal average speed isotropy. Since there are constraints on parameter b (it should be larger then zero and smaller then one) value minimizing the measure of the average speed isotropy can always be found for any given horizon. On Fig. 13 relation between the optimal value of b and the horizon radius is shown. As can be seen the optimal borrowed time parameter rapidly decreases and is more or less constant for horizon radii larger then 5. However, some trend can still be observed in this relation.

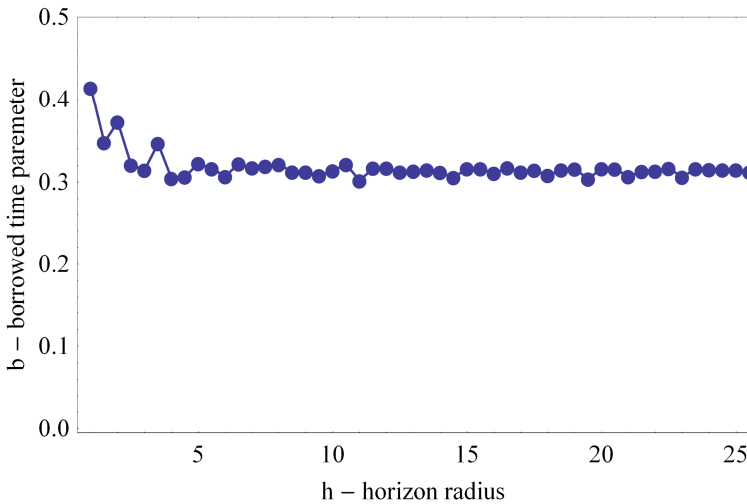


Fig. 13 A relation between the optimal borrowed time parameter mineralizing the measure of the average speed isotropy and the horizon radius

3.3 *A limit of the of the Optimal Borrowed Time Parameter at Infinite Horizon*

Unanswered lies the question what would be the optimal value of the borrowed time parameter if the horizon is located in infinity. For a large enough horizon radius size of a single cell can be neglected and it can be treated as a point. If we consider a horizon satisfying above assumptions with radius R for angles $\alpha \in [0, \pi/4)$ a time t needed to reach a cell through a optimal path P^{op} would be:

$$t = R \cos \alpha + bR \sin \alpha, \tag{5}$$

as an agent needs to make $R \cos \alpha$ steps and borrow time for $bR \sin \alpha$ diagonal moves. The average speed can be then parameterized with angle:

$$\bar{V}(\alpha) = \frac{R}{R \cos \alpha + bR \sin \alpha} = \frac{1}{\cos \alpha + b \sin \alpha}, \tag{6}$$

and used to calculate the angular dependence of the average speed deviation:

$$D(\alpha) = \frac{1 - \cos \alpha - b \sin \alpha}{\cos \alpha + b \sin \alpha}, \tag{7}$$

Sum in Eq. 3 becomes an integral for R approaching infinity, which using the symmetry property can be rewritten as:

$$I(\infty) = \int_0^{2\pi} D(\alpha) d\alpha = 8 \int_0^{\pi/4} D(\alpha) d\alpha = 8 \int_0^{\pi/4} \frac{1 - \cos \alpha - b \sin \alpha}{\cos \alpha + b \sin \alpha} d\alpha, \tag{8}$$

An integral in Eq. 8 can easily be calculated and minimized with regard to the borrowed time parameter resulting in $b \approx 0.309404$. For this value the angular isotropy and the measure of average speed isotropy is presented in Fig. 11 and Fig. 13. Such choice of the borrowed time parameter not only improved the average speed isotropy, but also caused the sign of the average speed deviation to be equally probable.

4 Concluding Remarks

This work presented approach to the grid type choice evaluation using the average speed isotropy. For the considered grid types the hexagonal grid showed highest isotropy. When geometry of the system does not posses any characteristic shape, usage of such grid should be considered. However, most of the human-made buildings and objects are by large percent square shaped. In such cases, usage of hexagonal grid is avoided as it does not represent such shapes well enough and square grid is used instead. As it was shown introducing the borrowed time concept the square lattice with Moore neighborhood can achieve high average speed isotropy and possibly reduce unwanted phenomena related to lower rotational symmetry of square grid. Thus, it is possible using this approach to have both accurate representation of square-like shapes and lower average speed anisotropy. This is especially important for applications such as pedestrian dynamics or transport simulations, where usually square lattice are used and transport characteristics of the system are of main interest.

It also should be noted that average speed isotropy is calculated under the assumption that agents always move by optimal paths, which can not always be true for non-sparse systems. In future works the triangular lattice will be investigated with the normal and extended neighborhoods, as well as how the characteristic of the considered system namely empirical angle distribution can be used for the grid type choice evaluation.

References

- [1] Chopard, B., Droz, M.: Cellular Automata Modeling of Physical Systems. Cambridge University Press, Alea-Saclay (2005)
- [2] Wolfram, S.: Cellular automaton fluids 1: Basic theory. *J. Stat. Phys.* 45(3/4) (1986)
- [3] Schonfisch, B.: Propagation of fronts in cellular automata. *Physica D: Nonlinear Phenomena* 80(4), 433–450 (1995)
- [4] Schonfisch, B.: Anisotropy in cellular automata. *BioSystems* (41), 29–41 (1997)
- [5] Kurrer, C., Schulten, K.: Propagation of chemical waves in discrete excitable media: anisotropic and isotropic wave fronts. In: Holden, M.M., Othmer, A.V., Othmer, H.G. (eds.) *Nonlinear Wave Processes in Excitable Media*. Plenum Press (1991)
- [6] Markus, M.: Dynamics of a cellular automaton with randomly distributed elements. In: Arino, A.D.E., Kimmel, O., Kimmel, M. (eds.) *Mathematical Population Dynamics*. Marcel Dekker (1991)
- [7] Nishiyama, A., Tokihiro, T.: Construction of an isotropic cellular automaton for a reaction-diffusion equation by means of a random walk. *J. Phys. Soc. Jpn.* 80, 054003 (2010)
- [8] Frish, U., Hasslacher, B., Pomeau, Y.: Lattice-gas automata for the navier-stokes equation. *Physical Review Letters* 56(14), 1505–1508 (1986)
- [9] Kirchner, A., Klupfel, H., Nishinari, K., Schadschneider, A., Schreckenberg, M.: Discretization effects and the influence of walking speed in cellular automata models for pedestrian dynamics. *J. Stat. Mech.* (2004)
- [10] Klupfel, H.: A Cellular Automaton Model for Crowd Movement and Egress Simulation. Dissertation, University Duisburg-Essen (2003)

Freshness Constraints in the RT Framework

Wojciech Pikulski and Krzysztof Sacha

Institute of Control and Computation Engineering,
Warsaw University of Technology, Warsaw, Poland
w.pikulski@elka.pw.edu.pl, k.sacha@ia.pw.edu.pl

Abstract. The work is focused on "time-domain" nonmonotonicity in the RT Framework, which is a Trust Management model. The freshness constraints propagation model that allows for credential revocation is presented. The solution introduces a freshness constraints that turn nonmonotonicity to be temporally monotonic. The freshness graph and propagation rules of freshness constraints are defined. The proposed solution allows policy authors to flexibly define constraints on policy level. Finally, the given model is evaluated against the real-life sample scenario.

1 Introduction

The RT Framework is a trust management model, which offers access control mechanisms dedicated for distributed systems. The decentralized design brings new problems, previously now known in the centralized one. The main characteristic of distributed systems is that the number of resources and users can be virtually unlimited. Additionally they can be located in different security domains, each controlling access to owned resources and issuing credentials to its users. Companies operating different security domains may set partnerships that require seamless collaboration of users and access to resources owned by each other. The RT framework addresses this problem by enabling delegation of authority between various domains.

Policy statements in the RT Framework are represented in form of credentials, which are digitally signed documents conveying security information. The framework utilizes them to represent roles needed to access a resource and roles that users belong to.

In a dynamic, distributed environment, it is normal to use credentials issued by different parties. In real-life scenarios it is common that some security statement has to be revoked, e.g. student id card may be cancelled when student is expelled from the university. The RT Framework originally does not address this problem. We adopted the notion of freshness presented in [1] and introduced freshness constraints propagation model that addresses credential revocation in the framework. An early version of the model is described in [2]. This chapter presents results of our further research in this area.

2 Related Work

The research is based on the RT Framework described in [3]. In [4] Chapin et. al. define requirements for trust management models and perform survey of proposed systems against them. One of the requirements is a monotonicity of the model, however, Li et. al. in [1] showed how certificate revocation can be implemented in a temporally monotonic manner.

Changyu et. al. in [5] propose a nonmonotonic trust management model called Shinren. Its concept differs from the RT Framework as it uses multi-value logic to allow for express negative statements in the policy whereas RT Framework is based on classical logic and permits to specify only positive expressions.

Another work that has similar aim that our research is the RT^R language presented in [6]. It allows to associate risk with each credential and performs risk aggregation along credential graph. When aggregated value exceeds defined threshold, further credentials are rejected. This model would be used to treat credential age as a risk and ignore all credentials for which aggregated risk exceeds threshold. However, it does not allow for credential status validation during processing an access request and making dynamic decisions based on verification result.

3 Business Problem

To illustrate the problem of credential revocation presented in the chapter, we introduce a sample real-life scenario. For this purpose, let imagine that there is an eStore selling mountaineering equipment that offers a discount for variety of people. Firstly, it is granted to long standing customers. Secondly, it is admitted to members of mountaineering clubs federated in National Mountaineering Association (NMA). Lastly, students who are members of Students Mountaineering Club (SMC) are also eligible for reduced price.

The eStore decided to identify long standing customers itself and delegate authority over other roles. Namely, SMC membership is delegated to the club, the student role is delegated to Accrediting Board of Universities (ABU), and mountaineering clubs memberships are delegated to NMA.

In the example scenario, we assume that there are two people: John and Adam. The former is a long standing customer. The latter is a student at the IT faculty of StateU University, which is accredited by the ABU. Adam is also member of SMC and Himalaya Club (HC). The latter is federated in NMA. The overall information can be expressed using the following credentials:

$$\begin{aligned}
 eStore.discount &\leftarrow eStore.longStandingCustomer \\
 eStore.discount &\leftarrow eStore.student \cap SMC.member \\
 eStore.discount &\leftarrow NMA.club.member \\
 eStore.longStandingCustomer &\leftarrow John \\
 eStore.student &\leftarrow ABU.university.student \\
 ABU.university &\leftarrow StateU
 \end{aligned}$$

$$\begin{aligned}
& \textit{StateU.student} \leftarrow \textit{StateU.faculty.student} \\
& \textit{StateU.faculty} \leftarrow \textit{IT} \\
& \textit{IT.student} \leftarrow \textit{Adam} \\
& \textit{SMC.member} \leftarrow \textit{Adam} \\
& \textit{NMA.club} \leftarrow \textit{HC} \\
& \textit{HC.member} \leftarrow \textit{Adam}
\end{aligned}$$

It is easy to see that both John and Adam are eligible for discount. Situation for John is straightforward. Adam case is more complicated, mainly because his eligibility to discount can be proved in two different ways. On the one hand, he is a student and a SMC member. On the other hand, a membership of NMA club does not require him to have student credential.

In the real-life scenario, the model should allow issuers to revoke credentials. For instance, when Adam fails to pay SMC membership fees or is expelled from university some credentials may be cancelled. Under such conditions, the eStore should not grant him a reduced price, but detect revocation and act appropriately.

4 Nonmonotonicity and Trust Management

Monotonicity of a security model means that if an access decision evaluates to true at some point in time, it should still be true if time lapses or additional credentials are introduced into the policy. There are two types of nonmonotonicity [2]. If time lapse makes access decision false, system is nonmonotonic in “time domain”. When addition of new credentials causes access decision false, system is nonmonotonic in “system size domain”. The former type is caused by credential revocation as cancelling a credential can cause that user will no longer be a member of specific role. However, an unaware acceptor may think that he still is. In this chapter we focus on “time domain” nonmonotonicity.

Li and Feigenbaum in [1] described how credential revocation can be implemented to be temporarily monotonic. They suggested to modify credential interpretation to the following: “at the time of t_f , certificate is valid from t_1 to t_2 ”. This is always true any time after t_f , even when credential was revoked. The t_1 parameter is not necessary, but it improves expressivity of the policy as issuers can create credentials before they become valid. Apart from new certificate interpretation, a notion of fresh time (t_f) of a certificate is introduced. Initially its value is set to certificate issue time ($t_f=t_o$). When acceptor ensures that certificate is still valid at a later time t_x , its fresh time is changed to it ($t_f=t_x$). Each acceptor defines a parameter Δt which states how old a certificate can be to be still accepted during authorisation evaluation, i.e. if $t_f \geq t_{now} - \Delta t$, certificate is regarded as valid. If the condition does not hold, authoriser needs to reject certificate or obtain proof that it has not been revoked. This chapter does not define how the certificate status is obtained as it is unimportant for the discussed topic. Authoriser may use different strategies to solve this problem, e.g. by checking Certificate Revocation List or using Online Certificate Status Protocol.

We call the Δt parameter a freshness constraint or requirement. If there is no ambiguity, we use term constraint. Setting Δt to small value implies frequent certificates validity checks. Defining big values limits validity verification but introduces risk of accepting revoked credential. The exact value of Δt depends on application and the level of risk it can accept.

After closer look at the idea of fresh time, one can find that Δt parameter should be specified not globally, but on more grained plane such as policy level. For example, some certificates can be treated as more vulnerable to revocation than others or some issuers can be treated as more trustworthy than others. When an user accesses a shared resource, the system checks available credentials and tries to decide whether an access should be granted. During this process, system should take into consideration freshness constraints defined for all credentials that make access decision evaluate to true. We call this problem a freshness constraint propagation and propose its formal model.

5 Freshness Constraints in the RT Framework

The RT Framework provides access control mechanisms for distributed systems, based on users and roles. The former are represented as set of entities which is denoted by *Entities*. A role is composed of an issuer, who is a member of *Entities* and a name of the role, which belongs to the set denoted by *RoleNames*.

There are four types of credentials in the RT Framework. Each credential has the following form $head \leftarrow body$. Both, head and body contain a *RoleExpression*.

1. Simple membership: $A.r \leftarrow D$. With this statement A asserts that D is a member of role $A.r$.
2. Simple inclusion: $A.r \leftarrow B.s$. Issuer A asserts that all members of $B.s$ are also members of $A.r$. This is a simple role delegation, since B can add new entities to $A.r$ role by adding them to $B.s$ role.
3. Linking inclusion: $A.r \leftarrow A.s.t$. Issuer A asserts that $A.r$ includes all members of $B.t$ role for each B that is member of $A.s$. This type of credential allows for authority delegation over $A.r$ to members of $A.s$, because each member B can permit access to the $A.r$ to other entity by granting it a $B.t$ role.
4. Intersection: $A.r \leftarrow C.s \cap D.t$. This credential allows A to assert that a member of $A.r$ is any entity that simultaneously is member of $C.s$ and $D.t$.

Authorisation procedure is based on a credential graph [7]. Its nodes are *RoleExpressions* that are parts of the credentials. Each credential corresponds to one normal edge in a graph. Additionally, there are derived edges that are consequence of linked roles and intersections. The notation $e_1 \Leftarrow e_2$ denotes an edge and $e_1 \ll e_2$ represents a path in the graph.

Definition 1. Credential Graph

Let C be a set of RT credentials. The basic credential graph G_C relative to C is defined as follows: the set of nodes $N_C = RoleExpressions$ and the set of edges E_C is the least set of edges over N_C that satisfies the following three closure properties:

1. If $A.r \leftarrow e \in C$ then $A.r \Leftarrow e \in E_C$. $A.r \Leftarrow e$ is called a credential edge.
2. If there exists a path $A.r_1 \ll B \in G_C$, then $A.r_1.r_2 \Leftarrow B.r_2 \in E_C$. A $A.r_1.r_2 \Leftarrow B.r_2$ is called a derived link edge, and the path $A.r_1 \ll B$ is a support set for this edge.
3. If $D \in N_C$ and $B_1.r_1 \cap B_2.r_2 \in N_C$, and there exist paths $B_1.r_1 \ll D$, and $B_2.r_2 \ll D$ in G_C , then $B_1.r_1 \cap B_2.r_2 \Leftarrow D \in E_C$. This is called a derived intersection edge, and $\{ B_1.r_1 \Leftarrow D, B_2.r_2 \Leftarrow D \}$ is a support set for this edge.

If an user should have access to a shared resource, there should exist a path linking a nodes representing resource and user. Such path with all support sets for derived edges is called a credential chain. It can be regarded as a proof that the user has access to the resource.

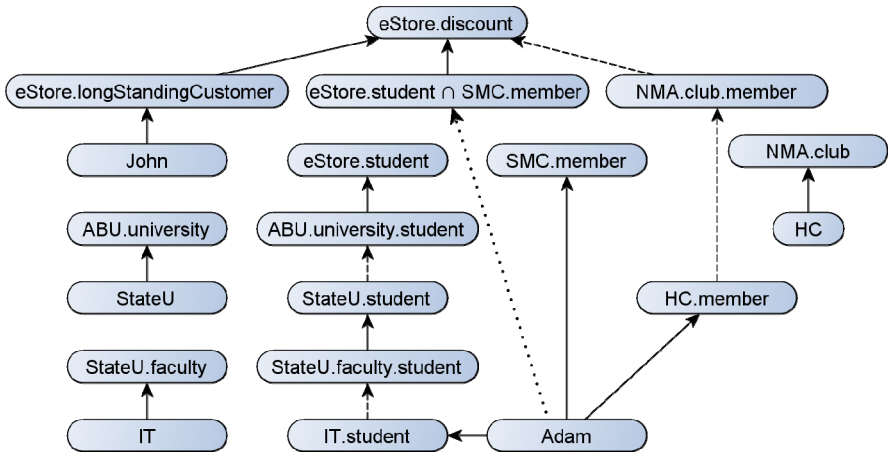


Fig. 1 Credential graph for sample scenario

Figure 1 presents the credential graph for the sample business scenario. It can be easily observed that both John and Adam have access to *eStore.discount* role. However, for John there exists only one credential chain whereas there exist two different credential chains for Adam. Normal edges are drew with solid arrays. Derived link edges and intersection edges are dashed and dotted respectively.

Assume that Adam orders some climbing equipment from eStore and he proceeds to the checkout. Also assume that policy in eStore does not has credential $eStore.discount \leftarrow NMA.club.member$. In such situation Adam has to be student and member of SMC. Let analyse what would happen if he get expelled from the university. In such circumstances he still owns his student credential that proves his untrue studying status. The credential probably would be issued for the whole semester so Adam would be able to act like a student for a couple of months before the credential expires.

The model presented in the chapter resolves this issue. The solution is to assign a parameter stating the maximum age to *eStore.discount* credential. Such parameter is called a freshness constraint or requirement.

To enhance expressiveness, policy authors should be able to define freshness constraints not only for roles representing accessed resources (i.e. *eStore.discount*)

but also for other roles (such as *SMC.member*). This may be desirable as policy authors may assume that some issuers might revoke credentials more frequently than others. When credential chain is being build, freshness requirements are propagated from resource role towards the user. This will ensure that all credentials in the chain will be fresh enough. The proposed model allows for defining freshness constraints on two levels of granularity.

1. Role definition level – used to define requirements per role basis, e.g. for *eStore.discount* role.
2. Entity level – defines constraints for all roles specified by given entity.

To allow for creating requirements depending on contextual information, a set of predicates $P=\{p_1, \dots, p_n\}$ is introduced. For example if eStore would like to define stricter constraint when order amount is over \$100, then set of predicates would be defines as follows: $P=\{order.amount > \$100\}$.

Definition 2. Freshness constraints

The freshness constraints are defined by f_c function: $f_c: D \times P \rightarrow \langle 0, \infty \rangle$, where $D = Entities \cup Roles \cup LinkedRoles$ and $P = [p_1, \dots, p_n]$ is a predicate vector where p_i is a logical value of one predicate.

Table 1 Freshness constraints for the sample scenario (f_c function)

D	Predicates	Δt
eStore		100
eStore.discount	order.amount > \$100	20
eStore.discount	-(order.amount > \$100)	70
ABU.university.student		180
SMC.member		30
NMA.club.member		60

The f_c function can be presented in a tabular form that contains only D members that have specified freshness constraint values. For all other members, an infinity is assumed.

To propagate freshness constraints along credential chain, a freshness graph is created. Figure 2 presents a freshness graph for sample scenario.

Definition 3. Freshness Graph

A freshness graph FG_C corresponds to credential graph G_C . The set of nodes are equal, i.e. $FN_C = N_C$ and set of edges FE_C is constructed as follows:

1. If $A.r \leftarrow e \in E_C$, then $A.r \Rightarrow e \in FE_C$, and it is called a freshness edge.
2. If $A.r_1.r_2 \leftarrow B.r_2 \in E_C$, then $A.r_1.r_2 \Rightarrow A.r_1 \in FE_C$, and $B \Rightarrow B.r_2 \in FE_C$, and they are called linked freshness edges.
3. If $B_1.r_1 \cap B_2.r_2 \leftarrow D \in E_C$, then $B_1.r_1 \cap B_2.r_2 \Rightarrow B_1.r_1 \in FE_C$ and $B_1.r_1 \cap B_2.r_2 \Rightarrow B_2.r_2 \in FE_C$, and they are called intersection freshness edges.

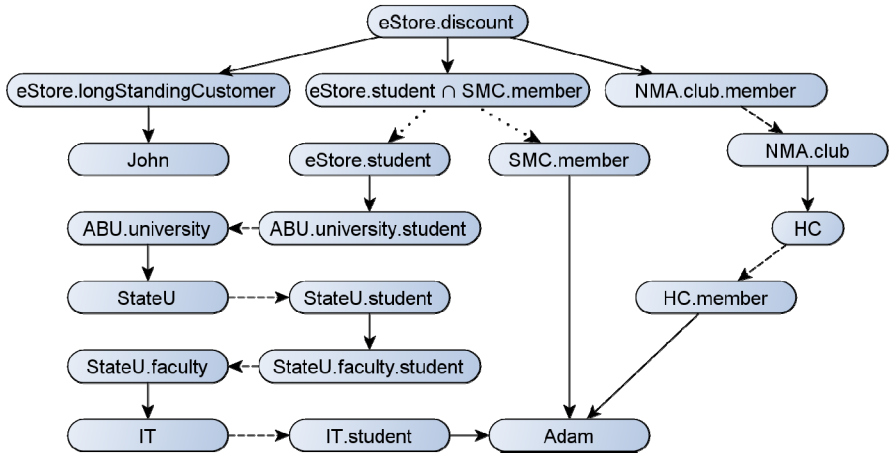


Fig. 2 Freshness graph for sample business solution

For each credential chain in the credential graph, there exists a corresponding path in the freshness graph. We call it a freshness chain. A freshness constraints propagation is performed for each such chain independently. This approach is different from mechanism described in [2]. In that work, constraints propagation was performed for all freshness graph in one step. However, this approach does not suit general cases where many credential chains can exist for a single access request. In such situation for each credential chain, the propagation process must be carried out individually. Because authorisation procedure in distributed systems can build many chains in parallel, the freshness requirements propagation process has to occur also in parallel. This topic exceeds the scope of this chapter and is not discussed in this work.

Policy authors define freshness constraints in form of f_c function. Those numbers are combined together to be used in the propagation process by using propagation operator ∇ defined as follows:

$$x \nabla y = \min(x, y)$$

A *calc* function is used to calculate for each *RoleExpression* a combined freshness constraints defined by policy author.

$$calc: RoleExpression \rightarrow \langle 0, \infty \rangle$$

$$\begin{aligned}
 calc(A) &= f_c(A) \\
 calc(A.r) &= f_c(A.r) \nabla calc(A) \\
 calc(A.r.s) &= f_c(A.r.s) \nabla calc(A.r) \\
 calc(A.r \cap B.s) &= \infty
 \end{aligned}$$

During access decision evaluation a credential chain is constructed. To enforce that all credentials forming a chain do not exceed the requirements for the role representing shared resource (e.g. *eStore.discount*) should propagate toward the

node corresponding to the user. To achieve that, for each constructed credential chain a *prop* function is applied to each vertex of corresponding freshness chain.

$$prop(n) = calc(n) \nabla (\bigvee_{\epsilon \in pre(n)} prop(\epsilon)),$$

where $n \in FN_C$, and $pre(n)$ is a set of predecessors of n

Please note that when node is an intersection, the *calc* function returns infinity and *prop* aggregates freshness constraints from the predecessors of the intersection. The intersection node has a number of successors equal to number of its elements. This strategy ensures that freshness requirements for one element will not propagate to other successors.

6 Case Study

In the sample scenario an eStore offers discount for students who are members of SMC or to people who belong to clubs federated in NMA.

Adam is eager climber who are studying and also belong to SMC and HC mountaineering clubs. Thus, he can obtain discount in eStore in multiple ways. Figure 3 presents various credential chains that prove that he is eligible for a reduced price. For each credential chain, a corresponding freshness chain with

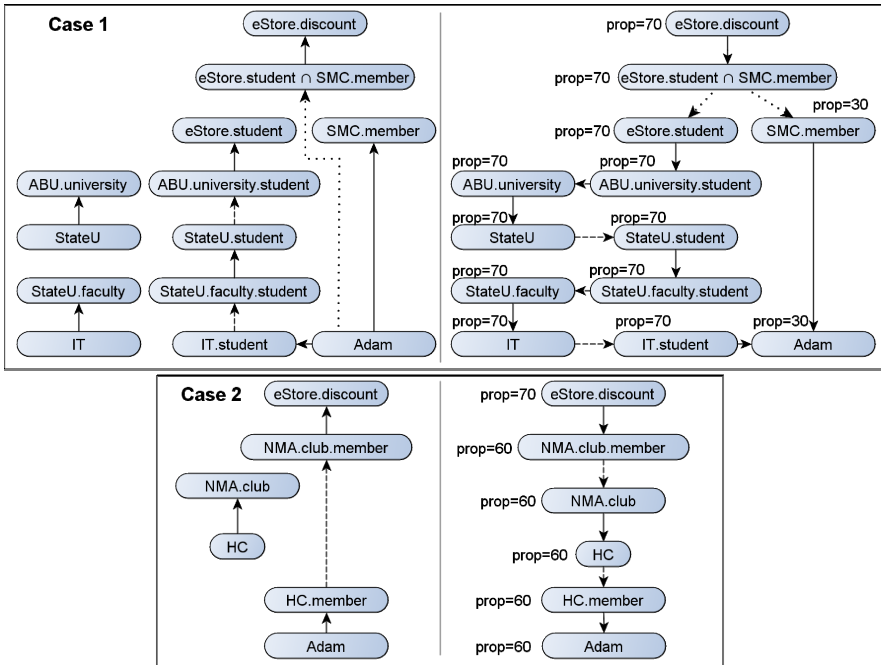


Fig. 3 Credential chains and corresponding freshness paths for sample scenario

propagated freshness constraints has been presented. For the purpose of the example, it was assumed that Adam order amount was below \$100. Therefore, the constraint for *eStore.discount* role is $calc(eStore.discount)=70$.

In Case 1, the constraint defined for *eStore.discount* is propagated along "left" side of the chain without modification. This is a consequence of the fact that for other roles in this part of the chain there have not been defined stricter constraints than for *eStore.discount*. However, this ensures that all this credentials must be at most 70 days old, so their age do not exceed the constraints of the *eStore.discount* role.

After constructing credential chain and carrying out the propagation process, each node of freshness chain has associated freshness constraint value calculated by *prop* function. This value is used to decide if credential is fresh enough to be valid or its validity should be checked. Because credential head contains its issuer, a *prop* value associated with node corresponding to the head should be used during verification.

Another place where verification should occur is authentication. When user wants to access a resource, he is authenticated by using public key certificate. A freshness of such document also must be checked. For this purpose a value assigned to the node representing user entity should be used.

This interpretation allows to make the following conclusions about the sample scenario. Each RT credential is represented by normal edge marked on graphs by solid line. For each such credential, a freshness constraint for its head should be used to verify its freshness. We must remember, that freshness requirements can depend on constructed credential chain. For example, in Case 1 credential *IT.student* ← *Adam* should be verified with value of $prop(IT.student)=70$, whereas for credential *SMC.member* ← *Adam* a value of $prop(SMC.member)=30$ should be used.

Similarly, an user public key certificate should be verified during user authentication. For Case 1 Adam's public key certificate must be at most $prop(Adam)=30$ days old but in the second case the constraint is more tolerable with value of $prop(Adam)=60$.

7 Summary and Conclusions

The chapter introduces a model that allows for credential revocation in the RT Framework. Policy authors can define freshness requirements on different levels of granularity. Those constraints are then propagated along constructed credential chain. Each credential is checked whether it is fresh enough to be used in authorisation process. If it exceeds requirements, its validity proof is obtained. The proposed model has been applied to a sample real-life scenario.

The future work will focus on examining relationship between credential graphs and freshness graphs. A connections among credential chains and freshness chains will also be investigated. The proposed model will be formally verified and extended to suit RT^T language and algorithms presented in [8].

References

- [1] Li, N., Feigenbaum, J.: Nonmonotonicity, User Interfaces, and Risk Assessment in Certificate Revocation. In: Syverson, P.F. (ed.) FC 2001. LNCS, vol. 2339, pp. 166–177. Springer, Heidelberg (2002)
- [2] Pikulski, W.: Nonmonotonicity in Trust Management. In: Szabó, R., Vidács, A. (eds.) EUNICE 2012. LNCS, vol. 7479, pp. 372–383. Springer, Heidelberg (2012)
- [3] Li, N., Mitchell, J., Winsborough, W.: Design of a Role-Based Trust-Management Framework. In: IEEE Symposium on Security and Privacy. IEEE Computer Society Press (2002)
- [4] Chapin, P.C., Skalka, C., Wang, X.S.: Authorization in trust management: Features and foundations. *ACM Comput. Surv.* 40 (2008)
- [5] Dong, C., Dulay, N.: Shinren: Non-monotonic Trust Management for Distributed Systems. In: Nishigaki, M., Jøsang, A., Murayama, Y., Marsh, S. (eds.) IFIPTM 2010. IFIP AICT, vol. 321, pp. 125–140. Springer, Heidelberg (2010)
- [6] Skalka, C., Wang, X.S., Chapin, P.C.: Risk management for distributed authorization. *Journal of Computer Security* 15, 447–489 (2007)
- [7] Czenko, M., Etalle, S., Li, D., Winsborough, W.H.: An Introduction to the Role Based Trust Management Framework RT. In: Aldini, A., Gorrieri, R. (eds.) FOSAD 2007. LNCS, vol. 4677, pp. 246–281. Springer, Heidelberg (2007)
- [8] Sacha, K.: Credential Chain Discovery in RTT Trust Management Language. In: Kottenko, I., Skormin, V. (eds.) MMM-ACNS 2010. LNCS, vol. 6258, pp. 195–208. Springer, Heidelberg (2010)

Transformational Modeling of BPMN Business Process in SOA Context*

Andrzej Ratkowski

Institute of Control and Computation Engineering
Warsaw University of Technology
Warsaw, Poland
a.ratkowski@elka.pw.edu.pl

Abstract. A transformational method for modeling business process is introduced in the following paper. Business process is described in Business Process Modeling Notation by using special constraints that embeds process in a SOA context. The transformational method supports a human designer by means of LOTOS language formalization.

1 Introduction

BPMN is standard *de facto* in business process modeling. Even though BPMN is a general purpose modeling notation, it can be also applied as a SOA orchestration language. In the current paper transformational method for BPMN process modeling in the SOA context is presented.

Having regard that business process modeling and modeling of orchestration of SOA services in practice usually is used to modify already existing processes and orchestration models. This is the motivation for concentrating on transformational modeling method rather on changing of existing processes than on modeling new processes from scratch.

Presented method is the development of transformation design of BPEL process in the SOA context [3].

1.1 BPMN in SOA Context

BPMN can be applied in two layers of SOA layered model [1]: business process layer and orchestration layer. BPMN diagrams in business process layer denote real-world processes that are executed by human actors or system actors. Diagrams in orchestration layer denote automatic processes executed in computer systems which control execution flows between SOA services. An example is shown in the fig. 1.

* This work was sponsored by the Polish Ministry of Science and Higher Education under grant number 5321/B/T02/2010/39.

A pool tagged PB is a process in the business process layer which is conducted in the real world. Actions X, Y and Z are tasks carried on by human actors. A pool tagged PU is in the orchestration layer. Activities A to D are invocations of single (atomic or composed) SOA services. Flows m1 and m2 denote execution of SOA orchestration process from business process layer.

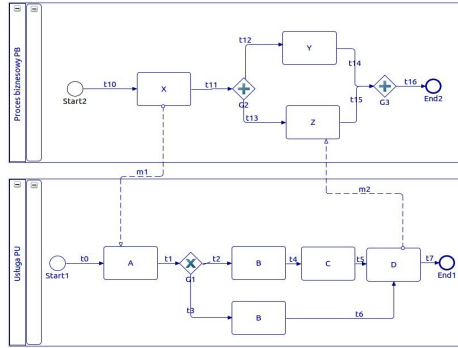


Fig. 1 Example application of BPMN in a SOA context

2 Transformational Method

Transformational method comprises four steps that are looped in a circular flow – fig 1. Modeling flow starts with *reference process* – a BPMN process that is initial version of modeled process. The designer conducts the following steps of the method:

1. **TRANSFORMATIONS** – reference process is subject of transformations. The method provides a set of standard transformations. As a result of this step we can identify a collection of *reference process variants* that are BPMN processes.
2. **BEHAVIOR VERIFICATION** – behavior of each variant is verified in comparison with reference process behavior. Each variant that is behaviorally not equivalent to the reference process is eliminated from modeling process (unless the human designer decide otherwise, that means approval of differences in its behavior).
3. **QUALITY EVALUATION** – quality of each approved variant is evaluated using quantified metrics. As a result of this step the best variant is obtained.
4. **VARIANT SELECTION** - in this step human designer decides whether the modeled process quality is acceptable. If so, modeling flow comes to an end, otherwise the best variant becomes new reference process and transformational method flow restarts.

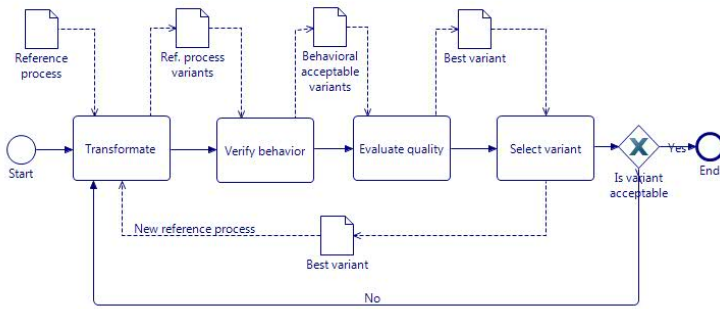


Fig. 2 Transformational method flow

2.1 Transformations

Transformational methods provides collection of atomic transformations that can be applied during modeling. Transformations are grouped in three categories:

- internal process activities manipulations: e.g. parallelization, serialization, permutation, activity removal etc.
- communication modification: e.g. change of communication mode from synchronous to asynchronous,
- processes manipulation: e.g. division of single process to two processes, moving activities between processes, merging processes etc.

Each transformation has premises/constraints and expected effect for modeled process. An example that shows description of parallelization transformation is in tab. 1. Human designer can also apply any transformation that is not in collection mentioned above , but in that situation that method does not support to anticipate an effect of transformation.

Finally as a result of transformations step of modeling method we can indicate a set of reference process variants.

Table 1 Parrallelization transformation

Parallelization	Process before transformation	Process after transformation
premise/ constraint	No dependence between activity X and Y – result of Y is not dependent on result of X	
effect	Better overall performance, shorter execution time	

2.2 Behavior Verification

Behavior of each reference process variant is examined by using LOTOS formalization. Both reference process and variant process are mapped to its specific LOTOS image. Mapping of BPMN process into its LOTOS image is named *DL* and is detailed introduced in section III.

If a variant is equivalent then is unconditionally qualified to next step. Otherwise, the verification method indicates differences in behavior and in that case the human designer decides that the variant can be modeled in the next step.

2.3 Quality Evaluation

Each variant that passed equivalence verification step is evaluated for quality metrics. Quality metrics are calculated according to properties of *DL* image of the variant. Metrics state quality in different criteria such as:

- performance – average response time, overall throughput, etc.
- process complexity and granularity – size, execution paths cardinality, average activities per process, etc.
- SOA compliance – level of mutual dependence of processes, etc.

Metrics for each variant and each criteria are calculated and constitute a quality vector, subsequently by using multicriteria optimization methods the best variant is selected.

2.4 Variant Selection

During the last step of modeling cycle a human designer decides whether the best variant is an acceptable solution. If so, modeling method ends and the best variant is a process result.

If the best variant is still not acceptable then best variant becomes new reference process and modeling cycle restarts.

3 BPMN to LOTOS Mapping

3.1 LOTOS Language

LOTOS is an ISO standardized language [2] designated to model concurrent and distributed communicating systems. It provides mathematical tools that help to analyze properties like equivalence between systems, bisimulation or execution tracing. We invented a mapping of BPMN elements to LOTOS expressions. The mapping is helpful to examine behavior of modeled process in behavior equivalence verification and can be used in calculating quantified properties of modeled processes. LOTOS language is briefly presented beneath:

- μ - action, represents every activity that occurs in the modeled system (sending a message, throwing a signal, and so on), action can be divided into two types: external observable g and unobservable (hidden) i
- B - behavior expression is a combination of actions, operators and other behavior expressions
- several operators that define rules of behavior:
 - ; - prefixing, $a;B$ means that the action a precedes behavior expression B ,
 - $[]$ - choice, $B1[]B2$ – means that means that either $B1$ or $B2$ is executed
 - $|||$ - parallel execution, $B1|||B2$ means that, $B1$ and $B2$ are executed in parallel
 - $|[x]|$ - parallel execution with synchronization on action x , $B1|[x]|B2$ means that, $B1$ and $B2$ are executed in parallel until first behavior expression reaches action x , the former expression stops and waits until the latter reaches action x ,
 - *exit* - operator that denotes successful execution of behavior expression
 - $>>$ - serial composition, $B1>>B2$ means that behavior expression $B2$ starts after successful execution of $B1$
- *process id_of _process[a1,a2,...] := B1 endproc* defines a process that represents some behavior expression $B1$. Arguments of a process $a1,a2,...$ are gates. Gates are actions that are interfaced to the defined process. Process can be instantiated by using *id_of _process[a1,a2,...]*

In (1) is presented simple example of LOTOS expression:

```

process Q[ a , b , c , d ] :=
    a ; ( b ; c ; exit [ ] b ; d ; exit )
endproc
    
```

(1)

The expression (1) should be interpreted as follows: on instantiation of the process Q at first occurs action a , then two actions: b then c or alternatively b then d .

The behavior of a LOTOS expression can be examined by analyzing *labeled transition system (LTS)* generated by that expression. LTS is a directed graph with edges that represents actions generated by LOTOS expression Vertice of LTS graph represents state that reaches LOTOS expression after occurring action on the edge that leads to the vertice. Rules of generating LTS for LOTOS expression is specified by LOTOS semantics[1].

3.2 Mapping Functions

BPMN process is mapped to LOTOS expression according to following general recipe:

- each BPMN element instance (activity, gate, flow etc.) corresponds to one LOTOS process definition; label of such process consists of label element instance and type of activity,

- if a BPMN element has outcomming flows then LOTOS definition contains instantiation of LOTOS processes corresponding to outcomming flows,
- LOTOS process that corresponds to a flow contains instantiation of LOTOS process *X*. Process *X* corresponds to BPMN element that is target of the flow.

Formally *DL(P)* and *BL(P)* functions are used to map BPMN to LOTOS. *DL(P)* function represents a LOTOS process definition corresponding to a fragment of BPMN process *P*. *DL* mapping uses another function denoted *BL(P)* that represents a LOTOS expression corresponding to a fragment of BPMN process.

In *DL(P)* and *BL(P)* functions are used following symbols denoting BPMN elements:

A – activities, each activity is a pair (*id, type*), where *id* is identifier and *type* \in {*activity, eventStart, eventIntermediate, eventEnd*}

G – gates, each gate is a pair (*id, type*), where *id* is identifier, *type* \in {*xor, event-based, parallel, ...*}

T – sequence flow, each flow is a triple (*from, to, id*), where *from* \in *A*, *to* \in *A*, *id* is identifier

C – message flow, each flow is a quadruple (*from, to, id, visibility*), where *from* \in *A*, *to* \in *A*, *id* is identifier, *visibility* \in {*internal, external*}

P – processes, *P* is a subset of cartesian product (*A* x *G* x *T* x *C*)

Mappings of basic elements of BPMN are presented in tables 2-5.

Table 2 Activity mapping


BPMN activity		AID – identifier of activity LABEL - label
DL(P)		<pre> DL({a \in A: a.id=AID})-> /* without outcomming flows*/ process act_AID[null] := LABEL;BL({t \in T: t.from=AID}) endproc /* alternatively with outcomming flow*/ process act_AID[null] := LABEL;BL({c \in C: c.from=AID});BL({t \in T:t.from=AID}) endproc + DL({t \in T: t.from=AID}) + DL({c \in C: c.from=AID}); </pre>
BL(P)		<pre> BL({a \in A: a.id=AID})->act_AID[null] </pre>

Table 3 Flow mapping


		<p><i>TID</i> – identifier of flow <i>TOID</i> – identifier of target element <i>TID1..TIDn</i> – identifiers of other outgoing flows beside <i>TID</i></p>
BPMN flow		
DL(P)		<p>DL({t ∈ T: t.id=TID})-> process trans_TID[null] := BL({a ∈ A: a.id=TOID}) endproc + DL({a ∈ A: a.id=TOID})</p>
BL(P)		<p><i>/*one outgoing flow*/</i> BL({t ∈ T: t.id=TID}) -> trans_TID[null] <i>/*alternatively for more than one outgoing flows*/</i> BL({T1 ∈ T: T1.from/T1.to=AID}) -> (trans_TID1[null][s_AID] ... trans_TIDn[null])</p>

Table 4 Gate mapping


BPMN flow		GID – identifier of gate
DL(P)		<p>DL({g ∈ G: g.id=GID})-> <i>/* XOR gate */</i> process gate_GID[null] := BL({t1 ∈ T: t1.from=GID}) [] ... BL({tn ∈ T: tn.from=GID }) endproc <i>/*alternatively parallel start gate*/</i> process gate_GID[null] := c BL({t1 ∈ T: t1.from=GID}) [s_GID] ... BL({tn ∈ T: tn.from=GID }) endproc <i>/* alternatively parallel end gate*/</i> process gate_GID[null] := BL({t ∈ T: t.from=GID}) endproc + DL({t ∈ T: t.from=GID})</p>
BL(P)		BL ({g ∈ G: g.id=GID})-> gate_GID[null]

Table 5 Process mapping

BPMN process	<i>PID</i> – identifier of process
DL(P)	$DL(\{p \in P: id=PID\}) \rightarrow$ process <i>proc_PID</i> [{ <i>C_{ext}.id</i> \subseteq C: <i>visibilty=external</i> }] := hide [{ <i>C_{int}</i> \subseteq C: <i>visibilty=internal</i> } in BL (<i>a</i> \in A: <i>type=eventStart</i>) endproc + DL (<i>a</i> \in A: <i>type=eventStart</i>)
BL(P)	BL (<i>p</i> \in P: <i>id=PID</i>) \rightarrow <i>proc_PID</i> [{ <i>C_{ext}</i> \subseteq C: <i>visibilty=external</i> }]

Result of mapping process *PB* in fig.1 is in listing 1, LTS for expression *proc_PB*[*m1,m2*] is in the fig. 3.

```

/*process*/
process proc_PB[m1,m2] := hide s_G3 in act_start2[null]          endproc
/*activities*/
process act_start2[null] := start2;trans_t10[null]              endproc
process act_X[null]    := X;m1;trans_t11[null]                  endproc
process act_Y[null]    := Y;trans_t14[null]                     endproc
process act_Z[null]    := Z;m2;trans_t15[null]                  endproc
process act_end2[null] := end2;stop                             endproc
process trans_t11[null] := gate_G2[null]                         endproc

/* gates */
process gate_G2[null] := trans_t12[null] |[s_G3]| trans_t13[null] endproc
process gate_G3[null] := trans_t16[null]                          endproc
/*flows*/
process trans_t10[null] := act_X[null]                            endproc
process trans_t12[null] := act_Y[null]                            endproc
process trans_t13[null] := act_Z[null]                            endproc
process trans_t14[null] := s_G3;gate_G3[null]                     endproc
process trans_t15[null] := s_G3;stop                             endproc
process trans_t16[null] := act_end2[null]                         endproc

```

Listing 1. MAPPING BL(PB)

Labels on edges of LTS graph corresponds to labels of BPMN activities and message flows. Paths of LTS graph are *execution paths*. In case of LTS generated for BL(P) function, set of all execution paths represents all possible orders of activities and message flows that can occur in the BPMN process.

Behavior of a BPMN process is interaction of the process with its environment, than can be observed outside. BPMN process can interact with the environment by sending and receiving messages by flows. So, behavior can be understood as an order of message flows. These message flows carry data, which is determined by order of activities occurring during execution process.

Above statements leads us to a conclusion that behavior of BPMN process is represented by LTS that is generated for expression, which in turn is a result of $BL(P)$ function.

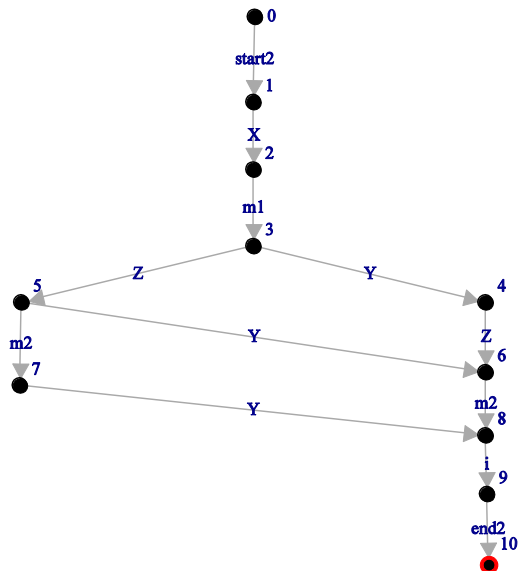


Fig. 3 LTS of $proc_PB[m1,m2]$ expression

3.3 Behavior Verification

BL(P) mapping of BPMN process can be utilized to verify behavior of BPMN processes during transformational modeling. Reference process and each reference process variant should be mapped to LOTOS by using BL(P) function. Afterwards, LTS of reference process and variant should be examined in order to find differences in execution paths. LOTOS provides us with special tools to do such examination - formally defined as *observational bisimulation* and *may* relationships[1]. These relationships indicate differences in execution paths of LTS and that differences can be interpreted as differences in behavior.

If a reference process LTS and one examined variant are in observational bisimulation relationship then the reference process is equivalent, in behavior, to the variant. Otherwise, when the reference process and the variant are inequivalent and execution paths, that that are missing or are in excess, represents behavior difference.

3.4 *Quality Metrics of BPMN Process*

LOTOS expression generated by BL(P) function can also be utilized to quantify quality properties of modeled process. Examples of quality metrics calculated in such way are enumerated below.

Performance of BPMN process can be estimated by transforming LOTOS expression to ET-LOTOS [1] – in order to enrich LOTOS action labels with time tags according to time characteristics of this actions (mean execution time, stochastic distribution, etc). Such expression generates LTS that can be treated as stochastic Markov chain, that can estimate mean execution time of BPMN process.

Complexity of BPMN process can be evaluated by size of LTS graph. Complexity size is number of vertices or edges of LTS.

SOA compliance of BPMN process can be estimated by number of LTS edges that correspond to message flows. Mutual dependence of processes increases together with number of message flows that connects processes.

4 Conclusions and Further Work

Transformational method presented in current paper is consistent and represents systematic approach to design BPMN processes in SOA environment. It provides human designer with formal tools to evaluate and infer behavior of modeled process.

It is planned to develop an application that could support human designer in applying proposed method flow and to automatize presented transformations: BL(P) function, behavior verification and quality metrics calculation.

Another planned features is integration of transformational method in process of SOA systems evolution presented in [4].

References

- [1] Marsan, M.A., Bianco, A., et al.: Integrating Performance Analysis in the Context of Lotos-Based Design. In: MASCOTS (1994)
- [2] ISO. Information Processing Systems: Open Systems Interconnection: LOTOS: A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour. ISO 8807:1989(E). International Organization for Standards (1989)
- [3] Ratkowski, A., Zalewski, A., Piech, B.: Transformational Design of Business Processes in Bpel Language. *e-Informatica Software Engineering Journal* 3(1) (2009), <http://www.e-informatyka.pl>
- [4] Zalewski, A., Kijas, S., Sokołowska, D.: Capturing Architecture Evolution with Maps of Architectural Decisions 2.0. In: Crnkovic, I., Gruhn, V., Book, M. (eds.) *ECSA 2011*. LNCS, vol. 6903, pp. 83–96. Springer, Heidelberg (2011)

Algorithmic and Information Aspects of the Generalized Transportation Problem for Linear Objects on a Two Dimensional Lattice

Anton Shmakov

Moscow Technical University of Communications and Informatics (MTUCI),
111024, Moscow, Aviamotornaya str. 8a
a.shmakov90@gmail.com

Abstract. This work is about logistic science and transportation problem. In first chapter author describe modern logistic and about its place in any knowledge domains. Second chapter is about common transportation problem and its modern particularity. Third chapter is about generalized transportation problem for two-dimensional space. In forth chapter author describe his solution of generalized transportation problem for linear objects. Fifth part is about futures of the work and transportation logistic at all.

1 Introduction

Quality of interaction between entities in any difficult system is important criterion of viability and development in it. The quantitative and qualitative characteristics of interactions had grown up with society development to essentially other level, which require their description and regulation. So there was [1] modern logistics – basis science about interactions between any subjects in all knowledge domains (Fig.1).

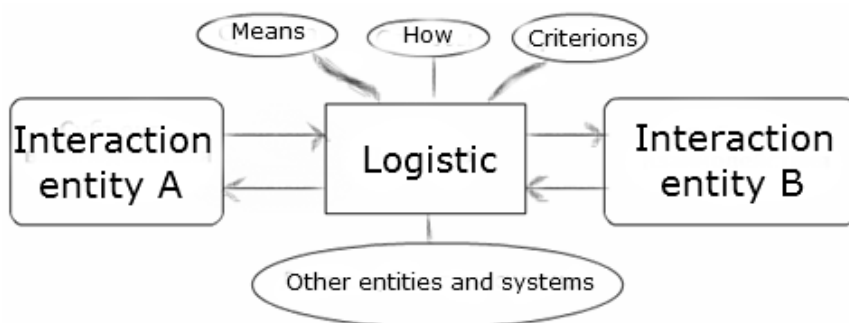


Fig. 1 Logistic in any system

The solution of the logistic task for one application scope can be used without excess conversions for other scope too. For example, the optimization problem of traffic flows, which described in this work, can be applied as to automobile transport networks of different level (from local problems of an enterprise, to global needs of city, region, and, may be, a world), and to optimization of network traffic, municipal support and so forth.

2 Transportation Problem

Transportation problem [2-3] is important part of supply chain problem [4-6]. Transportation problem was first formulated by Gaspard Monge [7], and major progress was obtained through the efforts of Leonid Kantorovich [8]. Volumes of transport flows increased much more significantly than capacity of space on which their relocation is carried out for the last 70-80 years. It is main difference between modern and historical transportation problem. Theoretically, it could be solved by evolution way in some domains like a communication networks, where new technologies and protocols could greatly expand network capacity. But in the case of road transport to provide the growing amount vehicles new roads in urban conditions seems impossible.

In other words, appearance of shared space in which independent transport flows are forced to interact among themselves is the main historical event for the transport task. This event caused appearance formulation of the transport task which described in the following chapter.

3 The Generalized Transport Task for Two-Dimensional Space

3.1 Formulation of the Problem

There is a limited two-dimensional area, partitioned by cells (measure units), that contains a group of suppliers (objects filling 1 cell of area at a single moment of time and have ability to move; single time moment is a time periods necessary for supplier to make 1 relocation for unit of area in one of basic directions), every of each should serve certain consumer (static final destination objects, that don't fill any area), i.e. every supplier matches only one consumer. The problem of optimal serving consumers should be solved, considering conflicts (at same moment of time at the same cell pretends more than one supplier).

The solution of such problem is set of some discrete functions, that represent paths of suppliers, i.e. set of sequential cells coordinates, that should pass the moving object to reach the destination point. For each of these functions operation and duration could be valued.

Path - function of time, each of discrete values of is a trajectory point.

Under optimization in this problem minimizing of total solution cost is understood, which in its turn, depends on spent operation and duration. A special case of optimization are problems of minimizing of operation and minimizing of duration (as usual, one is minimized at the expense of other).

3.2 Factors Affecting the Solution

In general case the difficulty of this problem and results of optimizations would be affected by these characteristics of input:

- Ratio of area filled by suppliers to area, available for moving.
- Existence, count and form of obstacles.
- Number of suppliers being able to move at the same moment (each period of time). Minimal value means alternate moves and maximal value is used for moving all producers at a time. Depending on used algorithm increasing of this parameter does not mean decreasing, and sometimes vice versa, leads to increasing required duration.

Connectivity – density of composite parts with respect to each other. The more average number of neighbors the moving objects have, the more connectivity presents.

Transposition – violation of direct geometry relevant of producer and consumer. An example of such transposition at Fig. 2.

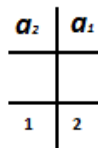


Fig. 2 Transposition of 1 and 2 of moving objects

4 Generalized Transportation Problem for Linear Objects

4.1 Formulation and Base Statements

The problem of linear located objects transportation into their linear located destination points on two-dimensional space (Fig. 3) should be solved. Arbitrary transposition (Fig. 4) of transportable objects and their destination points could be there. The solution of this task can be used for sorting of in parallel arriving boxes on a two-dimensional moving lattice.

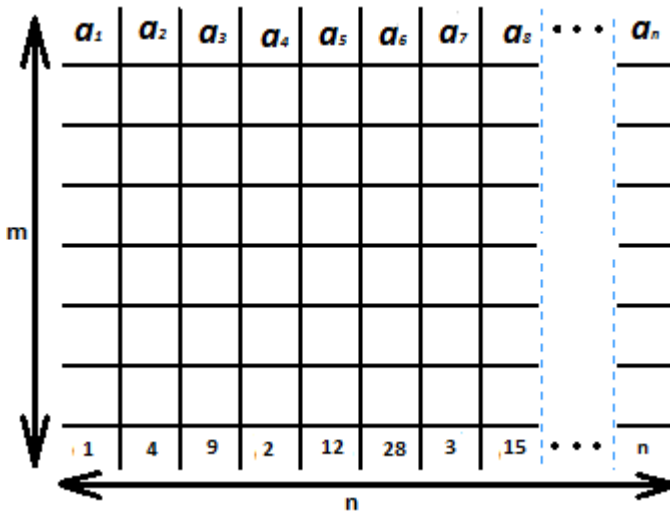


Fig. 3 Generalized view of problem

This task could have form of conformities matrix (1)

$$\begin{pmatrix} 1 & 2 & \dots & n \\ i_1 & i_2 & \dots & i_n \end{pmatrix} \tag{1}$$

Then solution F (2)

$$F : (1, \dots, n) \rightarrow (i_1, \dots, i_n) \tag{2}$$

represents a set of discrete functions of relocation (3)

$$f : (1) \rightarrow (i_1), f_2 : (2) \rightarrow (i_2), \dots, f_n : (n) \rightarrow (i_n) \tag{3}$$

which reflecting movement paths of objects to destination points.

Value of the general operation and the general time for F can be received on the following formulas (4, 5)

$$A = \sum_{i=1}^n A_i \tag{4}$$

$$T = \max(T_1, T_2, \dots, T_n) \tag{5}$$

The minimum operation can be evaluated for a situation of sequential relocation of suppliers (6)

$$A_{\min} = (m - 1)n + \sum_{k=1}^n |i_k - k| \tag{6}$$

The minimum time can be reached in case of such parallel movement in case of which the supplier from the maximum long path will have opportunity unobstructed (without the conflicts) relocation on all way (7)

$$T_{\min} = \max(|i_1 - 1|, |i_2 - 2|, \dots, |i_n - n|) + m - 1 \tag{7}$$

4.2 Algorithmic Approaches

In this section possible algorithmic approaches to be considered.

Sequence motion. To realize relocation of all suppliers one by one, using any search algorithm of a path [9-10] (like a Dijkstra’s algorithm, A*, JPS [11]) for bypass possible emerging obstacles. Such approach is effective for solving of the operation optimization task, but difficult to apply in practice. For this algorithm fairly following statement:

Statement: minimized operation from (6) could be achieved if

$$m' > 2 \text{ or } n' < 3 \tag{8}$$

is true (m and n represented at Fig 3). The duration in that case is equal to operation (9)

$$T'_{\min} = A'_{\min} = A_{\min} \tag{9}$$

Parallel motion. Implement moving of all objects in parallel, giving priority whenever conflict occurs using random values. Such method could be effective solving duration minimization problems, but gives unpredictable result on set of different problems.

Group motion. Divide figures to parts using conformity of moving units and destinations, than alternately relocate them. Such algorithm is quite effective for ordinary tasks. The more number of transpositions in input data the more operation and duration for moving would be spent. Variants of group transposition is possible – case of several nearby standing objects have equal shift to their destination. Number of groups belongs to interval [1..n]. There is an example of group transposition on Fig 4.

a_1	a_2	a_3	a_4	a_5
4	5	1	2	3

Fig. 4 Group transposition (subgroups [1,2,3] and [4,5])

Parallel group motion. Implement parallel motion of groups. In case when conflicts occur, they should be solved in a random way. Such algorithm actually is an improved version of sequence motion algorithm considering spent time:

$$\begin{cases} A'_{\min} = A_{\min} \\ T'_{\min} \geq T_{\min} \end{cases} \quad (10)$$

Parallel group motion using priorities. Implement parallel motion of all objects, previously rate them using priorities, using which build trajectories and solve conflicts. If formulation of problem matches condition (8), then such method would be most effective. Otherwise a duration optimization would be done in the prejudice of operation.

If in concrete formulation of problem satisfied condition (11):

$$m' > n' \quad (11)$$

Then

$$\begin{cases} T_{\min} \leq T'_{\min} \leq T_{\min} + 1 \\ A'_{\min} = A_{\min} \end{cases} \quad (12)$$

General structure of such algorithm is shown at Fig. 5.

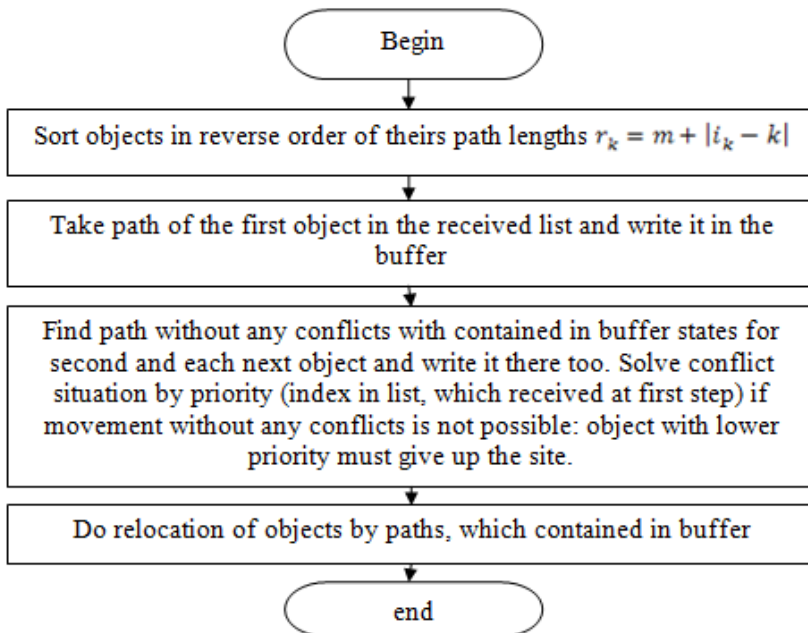


Fig. 5 General structure of parallel group motion using priorities algorithm

5 Problem for Dynamically Changing Systems

Real systems relative to synthetic have one important distinction - they are not constant and could not be fully described in advance. In real system at any moment of time an event [12] could occur, unprovided by static algorithm. The only way to describe such events using machine language - to bring probabilistic events, estimating their probability using theoretic and static analysis.

The final product that works with real systems should use maximum of foregone data, then apply corrections based on probabilistic expectations of different events. This product should be able to work in three modes:

- a) All states of system for all considering period of time are well known.
- b) Only some part of data is well known.
- c) No foregone data.

It means that obtained paths should be the most optimal for ideal conditions, probably optimal for real and final (using which moving from A to B is still possible) for worst/unexpected situations.

In a city the area images could be used (from the satellite, for example) to determine static and dynamic parts to build a road map with probabilistic obstacles: buildings and other barriers would be considered as obstacles with probability 1, other places that are used for traffic would be considered roads and statistic analysis should be used to determine their capacity. This data would base the system work, i.e. then could be considered constant.

Using modern technologies there is a possibility of determining current traffic using drivers gps-coordinates. This data should be used before actual start and during motion. If while moving appears unexpected situation, out of normal statistic data, and that prevents moving – a recalculation of trajectory should be done to match these statements:

- a) Entity reached its destination point.
- b) Groups of entities, that assumed similar trajectory, should not produce new traffic jam on a new road.

Today, there are several solutions that help the driver to determine the best route, but there is not good general-purpose solution to optimize the group motion of enterprise or of the city. So it's a good reason to promote solving of transportation problem that could help in many areas of live.

6 Conclusion

Results described in this work are only the first step in the solution of the tasks connected with real systems and actual for much of the main spheres of social life. Multiple-purpose solution of transportation problem can help with road traffic, data networks, pipeline flows, etc. The coherent mathematical theory in conjunction with the modern technologies can provide the solution of this problem and by that will give a powerful spur to world economy.

Acknowledgments. Special thanks to my supervisor, D.Sc., Professor Buslaev A.P., for formulation of the problem and useful suggestions.

References

- [1] Klaus, P., Müller, S.: Towards a Science of Logistics: Milestones along
- [2] Kolesnikov, A.V.: Transportation problem in modern mathematics. Seminar on traffic of the Russian Academy of Sciences under the leadership of Vladimir Kozlov (March 2, 2011)
- [3] Schrijver, A.: Combinatorial Optimization. Springer, Berlin (2003)
- [4] Blanchard, D.: Supply Chain Management. Best Practices
- [5] Rushton, A., Croucher, P., Baker, P.: The handbook of logistics & distribution management
- [6] Hugos, M.: Essentials of supply chain management
- [7] An Elementary Treatise on Statics with a Biographical Notice of the Author. Biddle, Philadelphia (1851)
- [8] Kantorovich, L.V.: Mathematical Methods of Organizing and Planning Production. Management Science 6(4) (1939, July 1960)
- [9] Cherkassky, B.V., Goldberg, A.V., Radzik, T.: Shortest paths algorithms: theory and experimental evaluation (1996)
- [10] Sanders, P.: Fast route planning. Google Tech Talk (March 23, 2009)
- [11] Harabor, D., Grastien, A.: Online Graph Pruning for Pathfinding on Grid Maps. In: Proceedings of the 25th National Conference on Artificial Intelligence (AAAI), San Francisco, USA (2011)
- [12] Lieu, H.: Traffic-Flow Theory. Public Roads (US Dept of Transportation) (January/February 1999)

Reliability Assessment of Supporting Satellite System EGNOS

Mirosław Siergiejczyk, Adam Rosiński, and Karolina Krzykowska

Warsaw University of Technology, Faculty of Transport
00-662 Warsaw, Koszykowa St. 75
{msi, adro, kkrzykowska}@wt.pw.edu.pl

Abstract. The paper presents the issues related to satellite navigation systems which can be supported by the EGNOS system. It provides basic information on the structure and operation of these systems, especially the EGNOS system. This enabled an analysis in terms of reliability of both the satellite navigation systems, as well as EGNOS. The results of values of probabilities staying the system in the highlighted states were achieved.

1 Introduction

Satellite navigation can be defined as a type of radio navigation, in which by emitted radio signals from artificial satellites the position of user can be defined. The concept of global navigation satellite system (GNSS) refers to a constellation of satellites providing signals from space that transmit position and time data of the object. By definition, GNSS has global range [1].

According to the manual published by ICAO - GNSS (Global Navigation Satellite System) is a collection of various satellite systems (GPS Navstar, GLONASS, Galileo in the future) and three basic support systems (ABAS, SBAS, GBAS). The only differences between systems such as GPS and GLONASS are based on signal coding methods used, reference systems in which the position is given, and the patterns of time. Existing satellite constellations, however, do not provide the four basic functions of the signal, namely: accuracy, continuity, credibility and reliability. Therefore, the structure is built to support operation of satellite navigation systems (WAAS, EGNOS, MSAS, LAAS, GAGAN, GRAS) [2]. Europe is launching its own independent satellite system, Galileo. When it becomes operational, Galileo will provide positioning services through a network of 30 satellites and related ground infrastructure. Galileo will be compatible with GPS and GLONASS.

2 Structure of Satellite Navigation Systems

All satellite navigation systems, both supporting and basic have the same structure, which is divided into three main parts (segments) [3]:

- space segment - a constellation of satellites orbiting the Earth in orbit, transmitting signals and navigation information;
- ground segment - service stations and supervision of the system;
- segment of users - information is provided to all users by the navigation system.

In addition, there are the following methods of supporting satellite navigation systems:

- monitoring by the receivers: RAIM - Receiver Autonomous Integrity Monitoring; ABAS - Aircraft Augmentation Based System; AAIM - Aircraft Autonomous Integrity Monitoring;
- GBAS ground support systems: DGPS; NDGPS (USA); SBAS satellite support systems: WAAS (USA); MSAS (Japan); EGNOS (Europe);
- pseudolites - ground equipment operating like additional satellites, information systems.

Satellite support systems (SBAS) such as EGNOS are regional solutions to improve the efficiency of GNSS. Satellite navigation system performance is evaluated according to four criteria:

1. Accuracy - refers to the difference between the actual and the measured position of the object, the speed and time of the receiver.
2. Integrity - refers to the capacity of the system in order to ensure, for example, alarms in case of irregularity in the position data.
3. Continuity - refers to the operation of the navigation system without interruption.
4. Availability - refers to a time unit in which the signal meets the criteria of accuracy, integrity and continuity.

EGNOS can increase the accuracy and reliability of the information correcting GPS signal measurement errors and providing information about the integrity of the signal [4]. These criteria must be met in order to provide the required navigation performance (RNP Required Navigation Performance).

3 Architecture of EGNOS System

EGNOS (European Geostationary Navigation Overlay Service) is a European system that supports GPS and GLONASS in the field of air transport, sea and road. It was designed by a group of ETG (European Tripartite Group), which includes: the European Space Agency ESA, the European Commission EC and the European Organisation for the Safety of Air Navigation EUROCONTROL [3]. The aim of the EGNOS is to monitor the reliability of the GPS and GLONASS and increase their accuracy by introducing correction data. The principle of operation is to receive GPS signals by ground reference stations to calculate the

position of the measurement error. Then, the calculated error is sent to the main reference station, which generates a differential correction. The amendment is sent by broadcasters to geostationary satellites.

The EGNOS system, like other satellite systems consists of three segments (space, ground, user). As an additional segment can be added assistive devices necessary for operation of the system, ie the PACF, the body responsible for the reliable operation of the system and ASQF - a unit whose mission is to provide the technical means necessary for the proper operation of EGNOS.

A space segment consists of three geostationary satellites (Inmarsat III AOR-E Inmarsat III IOR-W and ARTEMIS). They transmit data to the ground segment users [3].

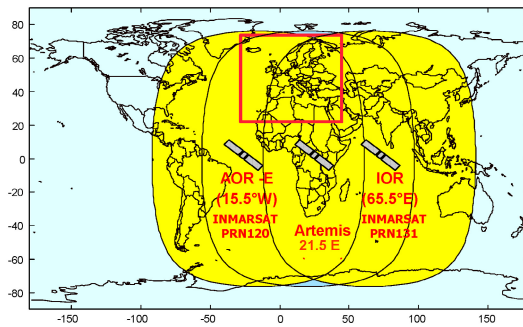


Fig. 1 Range of geostationary satellites of the EGNOS system

The EGNOS ground segment consists of [5]:

- 34 stations RIMS (Ranging and Integrity Monitoring Stations);
- 4 control centers MCC (Mission Control Centers);
- 6 stations NLES (Navigation Land Earth Stations);
- EWAN network (EGNOS Wide Area Network).

RIMS components are among others, receiving element (receiver and antenna), an atomic clock, the main computer and a network of FEE (Front End Equipment), which mediates the network connected to the EGNOS system. RIMS stations can also be considered from the point of their constituent channels. There are three channels A, B and C. The A channel data are used to calculate the data required to the EGNOS. The B channel data are used to verify the calculated messages via channel A. The C channel is designed to detect errors in the signal supplied by the GPS satellites. RIMS task is to collect data from the GPS satellites and their verification. EGNOS has 34 RIMS stations in channel A, 33 stations with channel B and 15 stations with channel C [5]. One of the stations with channels A and B is located in Warsaw, at the Center for Space Research. The appearance of the station antenna is shown in Fig. 2.

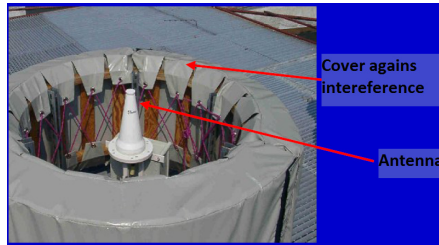


Fig. 2 Antenna of the RIMS station in CBK PAN in Warsaw

The tasks of the control centers MCC (Fig. 3.) include managing the entire system, EGNOS, processing of data received from other components, verification and archiving of data, calculation of the necessary amendments to the measured signals. In addition, the MCC can be divided into components of the CCF (Central Control Facility) and CPF (Central Processing Facility).

NLES stations, which are nevertheless an important component of the EGNOS ground segment, are responsible for carrying out the modulation, coding and signal transmission to the EGNOS geostationary satellites [3]. Both sending a signal created by EGNOS, as well as its subsequent reception makes NLES stations also involved in checking the credibility of the whole system [5].

The tasks that are put on the EWAN network should connect geographically distant EGNOS ground components into one functional unit. EWAN uses UDP / IP, TCP / IP and SNMP.

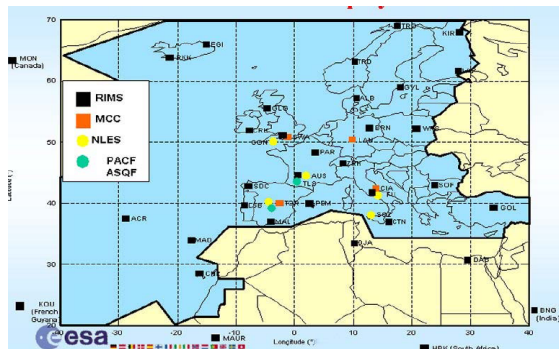


Fig. 3 The components of the EGNOS ground segment layout

The user segment consists of all navigation GPS / GLONASS, which have built-in cooperation with EGNOS, as shown in Fig. 4.

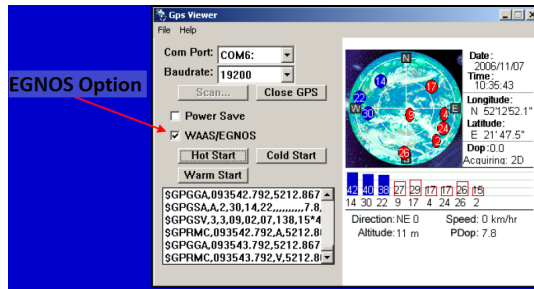


Fig. 4 GNSS signal receiver with built in function of cooperation with EGNOS

After concretizing EGNOS architecture and its components, the principle tasks of operation of the system can be specified. It is based on the fact that the measurements are made by the RIMS stations - the stations MCC are calculating amendments to the measured signals for all observed satellites and the signal delay of the path from the satellite to the RIMS. Then, these data are transmitted to geostationary satellites through NLES stations. This transmission is performed on the GPS L1 frequency (1575.42 MHz) prior to the signal modulation and coding [3].

Achieving operational activities of the EGNOS system will undoubtedly increase the range of use of the currently operated satellite navigation systems [5]. Before the EGNOS system, however, poses some additional requirements, such as increasing the number of geostationary satellites, the removal of restrictions on the range of EGNOS, and verification plans to modernize GPS and GLONASS systems in the context of data integrity.

4 Analysis of Satellite Navigation System in Terms of Reliability

Analyzing any of the satellite navigation system, it can be said that it has a mixed reliability structure [6-11]. Damage of the ground segment of the system moves it from the state of full ability $R_0(t)$ to the state of unreliability $Q_B(t)$. Damage of one artificial satellites in the space segment causes the transition from full ability $R_0(t)$ to a state of partial ability $Q_{ZB}(t)$ [12,13,14]. The more satellites are visible to the receiver located in the user segment, the more states of partial ability. Of course, it must be keep in mind that required minimal number of satellites to determine the location of the object is 3. Figure 5 shows the relationships in the present satellite navigation system in terms of reliability [15].

In consideration for electronic devices constant failure rate was adopted and, therefore, for further analysis - the exponential distribution.

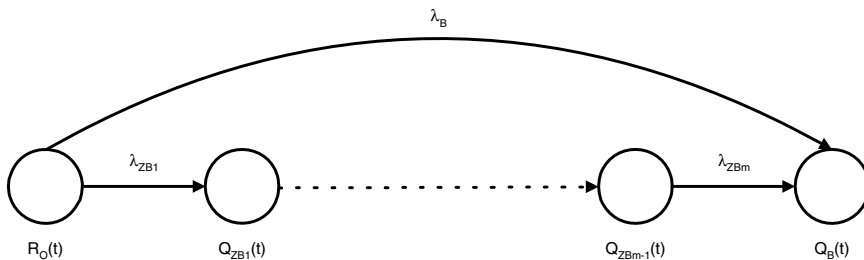


Fig. 5 Relations in satellite navigation system

$R_O(t)$ – function of probability of system staying in the state of full ability,
 $Q_{ZB}(t)$ – function of probability of system staying in the state of partial ability,
 $Q_B(t)$ – function of probability of system staying in the state of unreliability,
 λ_B – intensity of transition of the ground segment,
 λ_{ZBi} – intensity of transition of the satellites.

It is possible to determine the probabilities of staying in the respective states: full ability R_O , the partial ability Q_{ZB} and unreliability Q_B by the considered system.

The presented analysis allows to determine the probabilities of the proper functioning of the satellite navigation system. It does not cover the possibility of the use of EGNOS as a system to support the functioning of the satellite navigation system. This aspect (eg. due to the safety of passengers) is very important in relation to the location of vehicles and objects in the field of air transport, road and sea [16,17,18].

In carrying out the operation of the satellite navigation system assisted by the EGNOS system, the relationships can be illustrated in this system, in terms of reliability, as shown in Fig. 6.

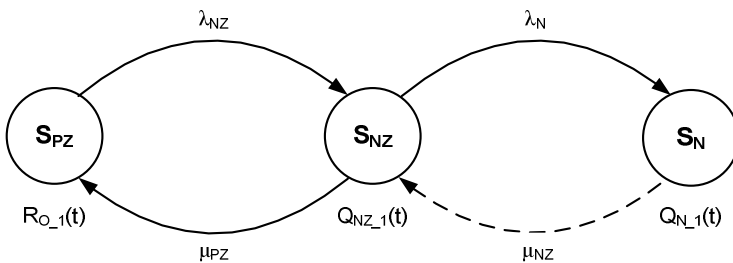


Fig. 6 Relations in the satellite navigation system supported by EGNOS

$R_{O,1}(t)$ – function of probability of system staying in the state of full ability,
 $Q_{NZ,1}(t)$ – function of probability of system staying in the state of partial functionality,
 $Q_{N,1}(t)$ – function of probability of system staying in the state of unreliability,
 λ_{NZ} – intensity of transition from the state of full ability to partial functionality,
 μ_{PZ} – intensity of transition from the state of partial functionality to full ability,
 λ_N – intensity of transition from the state of partial functionality to unreliability,
 μ_{NZ} – intensity of transition from the state of unreliability to partial functionality.

By turning off the EGNOS system in the receiver (Fig. 6.), the transition from the state of full ability to partial functionality. Turning on the EGNOS system in the receiver transit it back to the state of full ability. If there is a state S_{NZ} (consisting of partial functionality of EGNOS) – satellite navigation system failure means that the system is back to the state of unreliability. Returning from state S_n to S_{nz} is possible, but not taken into account in further considerations.

When performing mathematical analysis (eg using the Kolmogorov-Chapman equation) there are some equations allowing to designate probabilities of system being in states of full ability $R_{O_{-1}}$ (1), partial functionality $Q_{NZ_{-1}}$ (2) and unreliability $Q_{N_{-1}}$ (3).

$$R_{O_{-1}}(t) = \left[\begin{aligned} &\cos\left(\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2} \cdot \frac{t}{2}\right) + \\ &+ \frac{\mu_{PZ} + \lambda_N - \lambda_{NZ}}{\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2}} \cdot \\ &\cdot \sin\left(\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2} \cdot \frac{t}{2}\right) \end{aligned} \right] \cdot \exp\left[-\left(\frac{\lambda_{NZ} + \mu_{PZ} + \lambda_N}{2}\right) \cdot t\right] \tag{1}$$

$$Q_{NZ_{-1}}(t) = \frac{2 \cdot \lambda_{NZ}}{\sqrt{2 \cdot \lambda_{NZ} \cdot \lambda_N - 2 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2}} \cdot \sin\left(\sqrt{2 \cdot \lambda_{NZ} \cdot \lambda_N - 2 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2} \cdot \frac{t}{2}\right) \cdot \exp\left[-\left(\frac{\lambda_{NZ} + \mu_{PZ} + \lambda_N}{2}\right) \cdot t\right] \tag{2}$$

$$Q_{N_{-1}}(t) = 1 - \left[\begin{aligned} &\cos\left(\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2} \cdot \frac{t}{2}\right) + \\ &+ \frac{\mu_{PZ} + \lambda_N + \lambda_{NZ}}{\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2}} \cdot \\ &\cdot \sin\left(\sqrt{2 \cdot \lambda_{NZ} \cdot (\mu_{PZ} + \lambda_N) - 4 \cdot \mu_{PZ} \cdot \lambda_{NZ} - \lambda_{NZ}^2 - (\mu_{PZ} + \lambda_N)^2} \cdot \frac{t}{2}\right) \end{aligned} \right] \cdot \exp\left[-\left(\frac{\lambda_{NZ} + \mu_{PZ} + \lambda_N}{2}\right) \cdot t\right] \tag{3}$$

5 Modeling the Reliability of Satellite Navigation System EGNOS

Methods and computer-simulation studies make it possible to determine the impact of relatively rapid change of reliability and operational indicators of individual items on the reliability of the whole system.

Using computer support can be carried out, among others, analysis of the impact of recovery time of the satellite navigation system EGNOS to full ability R_{O_1} , partial functionality Q_{NZ_1} and unreliability Q_{N_1} . It is shown in the following example.

Example

Assume the following values for the system analyzed:

- Time of research – 1 year:

$$t = 8760[h]$$

- Unbreakability of EGNOS system:

$$R_{NZ}(t) = 0,999$$

- Reliability of the satellite navigation system:

$$R_N(t) = 0,9999$$

Knowing the unbreakability value $R_{NZ}(t)$, it can be assumed the intensity of transition from the state of full ability to the state of partial functionality. Assuming the simplest model of an exponential distribution of ability time, we may use the following relationship:

$$R_{NZ}(t) = e^{-\lambda_{NZ}t} \text{ for } t \geq 0$$

so

$$\lambda_{NZ} = -\frac{\ln R_{NZ}(t)}{t}$$

For $t = 8760[h]$ and $R_{NZ}(t) = 0,999$ we get:

$$\lambda_{NZ} = -\frac{\ln R_{NZ}(t)}{t} = -\frac{\ln 0,999}{8760} = 1,142124 \cdot 10^{-7} \left[\frac{1}{h} \right]$$

Knowing the unbreakability value $R_N(t)$, it can be assumed the intensity of transition from the state of partial functionality to the state of unreliability. Assuming the simplest model of an exponential distribution, we may use the following relationship:

$$R_N(t) = e^{-\lambda_N t} \text{ for } t \geq 0$$

so

$$\lambda_N = -\frac{\ln R_N(t)}{t}$$

For $t = 8760[h]$ and $R_N(t) = 0,9999$ we get:

$$\lambda_N = -\frac{\ln R_N(t)}{t} = -\frac{\ln 0,9999}{8760} = 1,14161 \cdot 10^{-8} \left[\frac{1}{h} \right]$$

The intensity of transition from the state of partial functionality to the state of full ability μ_{PZ} is, as well known, (in case of exponential distribution) – time reversal t_{PZ} :

$$\mu_{PZ} = \frac{1}{t_{PZ}}$$

If we assume that time of get back to the state of full ability t_{PZ} can be between $t_{PZ} \in \langle 12;168 \rangle [h]$, then probability of being by the the analyzed sytem in the different state can be shown like at the charts below 7, 8.

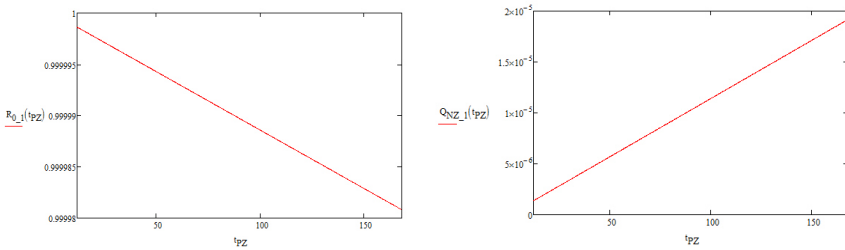


Fig. 7 Relation between probability of system being in the state of full ability R_{O_1} , Q_{NZ_1} in function of time t_{PZ}

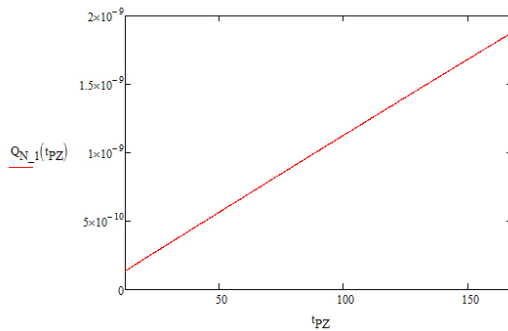


Fig. 8 Relation between probability of system being in the state of unreliability Q_{N_1} in function of time t_{PZ}

6 Summary

The presented process of satellite navigation systems analysis allows us to determine the level of reliability of the proposed system. This is possible by using the required number of satellites that can ensure an adequate level of reliability indicators.

The paper presents an analysis of satellite navigation system EGNOS with particular emphasis on the impact of full-time return to the values of probabilities being the system in states of full ability R_{O_1} , partial functionality and unreliability Q_{N_1} Q_{NZ_1} . Improving the reliability of the system utility can be done by reducing the recovery time of suitability. In a further study of this issue should be sought to determine the relationship between financial inputs, sometimes associated with the restoration of full ability, and the probability of system staying in the highlighted technical conditions.

References

- [1] EGNOS, What is GNSS? (2011), <http://www.egnos-portal.eu> (accessed December 27, 2012)
- [2] Narkiewicz, J.: GPS i inne satelitarne systemy nawigacyjne, Warszawa (2007)
- [3] Januszewski, J.: Systemy satelitarne GPS Galileo i inne, Warszawa (2007)
- [4] European Organisation for the Safety of Air Navigation EUROCONTROL, NAV-GNSS Global Navigation Satellite System Training Provided by the IANS ATM Unit, Luxembourg, Kirchberg (2007)
- [5] ESA, European Geostationary Navigation Overlay Service (2012), <http://www.cbk.waw.pl> (accessed December 27, 2012)
- [6] Będkowski, L., Dąbrowski, T.: The basis of exploitation, part II: The basis of operational reliability. Wojskowa Akademia Techniczna, Warsaw (2006)
- [7] Epstein, B., Weissman, I.: Mathematical models for systems reliability. CRC Press / Taylor & Francis Group (2008)
- [8] Kołowrocki, K., Soszyńska-Budny, J.: Reliability and safety of complex technical systems and processes. Springer, London (2011)
- [9] Levitin, G.: The Universal Generating Function in Reliability Analysis and Optimization. Springer, London (2005)
- [10] Stapelberg, R.F.: Handbook of Reliability, Availability, Maintainability and Safety in Engineering Design. Springer, London (2009)
- [11] Zamojski, W. (ed.): Reliability and Maintenance of Systems. Publisher of Wrocław University of Technology, Wrocław (1981)
- [12] Rosiński, A.: Reliability analysis of the electronic protection systems with mixed – three branches reliability structure. In: Proc. International Conference European Safety and Reliability (ESREL 2009), Prague, Czech Republic, pp. 1637–1641 (2009)
- [13] Rosiński, A.: Reliability analysis of the electronic protection systems with mixed m-branches reliability structure. In: Proc. International Conference European Safety and Reliability (ESREL 2011), Troyes, France (2011)
- [14] Ważyńska-Fiok, K., Jaźwiński, J.: Reliability of technical systems. PWN, Warsaw (1990)

- [15] Rosiński, A.: Design of the electronic protection systems with utilization of the method of analysis of reliability structures. In: Proc. Nineteenth International Conference on Systems Engineering (ICSEng 2008), Las Vegas, USA, pp. 421–426 (2008)
- [16] Dyduch, J., Paś, J., Rosiński, A.: The basic of the exploitation of transport electronic systems. Technical University of Radom, Radom (2011)
- [17] Siergiejczyk, M.: Maintenance Effectiveness of Transport Telematics Systems. Transport Series, vol. (67). Scientific Works of the Warsaw University of Technology, Warsaw (2009)
- [18] Wawrzyński, W., Siergiejczyk, M., et al.: Final Report on Grant KBN 5T12C 066 25. Methods for Using Telematic Measures to Support Realisation of Transport Tasks, Supervisor: Associate Professor Ph.D. D.Sc. W. Wawrzyński, Warsaw (2007)

Vbam – Byzantine Atomic Multicast in LAN Based on Virtualization Technology

Marcelo Ribeiro Xavier Silva, Lau Cheuk Lung,
Leandro Quibem Magnabosco, and Luciana de Oliveira Rech

Computer Science – Federal University of Santa Catarina – Florianópolis, Brazil
marcelo.r.x.s@posgrad.ufsc.br,
{lau.lung,luciana.rech}@ufsc.br,
leandroqm@gmail.com

Abstract. This work presents a BFT Atomic Multicast Protocol (Vbam) whose algorithm manages to implement a reliable consensus service with only $2f + 1$ servers using only common technologies, such as virtualization and data sharing abstractions. In order to achieve these goals, we chose to adopt a hybrid model, which means it has different assumptions between components regarding synchrony, and two different local area networks (LANs), a payload LAN and a separated LAN where message ordering happens.

1 Introduction

The difficulty to build distributed systems can be strongly reduced by relying on group communication primitives such as atomic (or totally ordered) multicast [1]. The atomic multicast ensures that messages sent to a set of processes are delivered by all these processes in the same order. Atomic multicast has many important applications such as clock synchronization, CSCW, distributed shared memory, database replication [2], [3], [4] and basis the state machine approach, the main component of many fault-tolerant systems [5], [6], [7], [8].

There exists a considerable amount of literature on total order multicast, and many algorithms, following various approaches, have been proposed to solve this problem. However, in most cases, these algorithms consider system models subject only to crash faults [1], [9]. Very few address also byzantine/arbitrary faults [10], [7]. In general, these relies on a consensus algorithm to agree on messages ordering and need $3f + 1$ processes involved in the agreement. There are some works that separate the consensus from the agreement problem creating a consensus service [11], [12].

We present Vbam, a byzantine fault tolerant consensus service for atomic multicast messages. The model and architecture that we propose needs only $2f + 1$ servers compounding the consensus service and is based on a hybrid model where there exist variation, from component to component, on the assumptions of synchrony and presence/severity of failures [13], [14], [15]. In this model, there is a

payload LAN in which the clients communicate with the consensus service, and an inviolable LAN where the servers do the ordering. In our proposal we contribute with an improvement to make the consensus service [11] byzantine fault tolerant with low resilience (only $2f + 1$ servers).

2 Related Work

The atomic multicast problem has been widely addressed in the past decades [2], [10], [14], [3], [9], [7]. In the majority, the approaches consider that the processes can only crash, i.e., not acting in arbitrary/byzantine manner.

In [10], it is presented Rampart for reliable and atomic multicast in systems with byzantine faults. The algorithm is based on a group membership service, which requires that at least one third of all processes in the current view reach an agreement on the exclusion of some process from the group. The atomic multicast is done by some member of the group, called sequencer, which determines the order for the messages in the current view. In the next view, another sequencer is chosen by some deterministic algorithm. Rampart assumes an asynchronous system model with reliable FIFO channels, and a public key infrastructure known by every process. With the assumption of authenticated communication channels the integrity of messages between two non-Byzantine processes is always guaranteed.

Guerraoui and Schiper proposed in 2001 the generic consensus service [11] for solving agreement problems, including the atomic multicast. This is the base for our proposal. Their model considers a crash-only environment with a consensus service that separates the consensus from the agreement problem to be solved. The system requires a perfect failure detector (that basis the consensus server) and the resilience is a tradeoff with performance, varying depending on the necessity.

In 2006 Correia and Veríssimo showed a transformation from consensus to atomic broadcast [7]. The system model presented assumes a byzantine environment in which up to $f = \lfloor (n - 1)/3 \rfloor$ faults are tolerated. The authors implement a multi-valued consensus protocol on top of a randomized binary consensus and a reliable broadcast protocol. The atomic multicast protocol is designed as successive transformations from the consensus protocol. The atomic multicast is done by the use of a hash vector. Each process of the system proposes values to the consensus vector (that is a vector with the hashes of the messages). The vector consensus protocol decides on a vector X_i with at least $2f + 1$ vectors H from different processes. In the sequence the messages are stored in a set to be atomically delivered in a pre-established order.

In 2010 Pieri et al proposed an extension of the generic consensus service [11] for byzantine environments [12]. The system model proposed has $n_c = 3f_c + 1$ clients and $n_s = 2f_s + 1$ servers, and they make use of virtual machines to provide the generic consensus service. The atomic consensus starts whenever one of the processes called initiator reliably multicast a message m_i to the clients set. Upon receiving the message m_i , each client sends a proposal to the generic consensus service for m_i . When the servers receive $n_c - f_c$ proposals from clients to the same

consensus instance, each server start a consensus protocol. Then, the result of this protocol is relayed to the clients. The importance of this work comes in means of making the generic consensus available for byzantine environments. However, it limits the size of the clients set. Furthermore the system has to deal with the faulty clients, lowering the system resilience. This is acceptable when dealing with generic agreement problems, but for solving the atomic multicast it is expensive.

Table 1 Comparison on evaluated atomic multicast protocols properties

	<i>Rampart</i> [10]	<i>Guerraoui and Schiper</i> [11]	<i>Correia and Verissimo</i> [7]	<i>Pieri et al</i> [12]	<i>Vbam</i>
Resilience for atomic multicast	$3f + 1$	-	$3f + 1$	$3f_c + 1 + 2f_s + 1$	$2f + 1$
Communication steps	6	5	-	5	4
Messages exchanged	$6n - 6$	$3n_c + 2n_c - 3$	$18n^2 + 13n + 1 + 16n^2f + 10nf$	$2(n_s^2 + 3n_c - n_s - 1)$	$3n_s^2 - n_s + n_c + 1$
Tolerated faults	Byzantine	Crash	Byzantine	Byzantine	Byzant

3 System Model and Architecture

The system model is hybrid [14], which is where assumptions of synchrony and presence/severity of failures vary from component to component [13], [14]. In our model, we consider different assumptions for the subsystems running in host, than for the ones running in the guest of the virtual machines that composes the system. In this model, $C = \{c_1, c_2, c_3, \dots\}$ is a set that contains a finite number of client processes and $S = \{s_1, s_2, s_3, \dots, s_n\}$ representing a set of servers with n elements that compound the consensus service.

The Figure 1, each consensus service server is hosted by a virtual machine. Each server physical machine has one, and only one, virtual machine as its guest (see Figure 1). The process failure model admits a finite number of clients and up to $f \leq [(n - 1)/2]$ servers incurring failure based on its specifications, presenting byzantine faults [17]: the faulty processes that arbitrary detour from its specification can stop, omit sending or receiving messages, send wrong messages or have any non-specified behavior. However, we assume independence of faults, in other words, the probability of a process having a fault is independent of another fault in any other process. This is possible in practice by the extensive use of diversity (different hardware, operational systems, virtual machines, databases, programming languages, etc.) [18].

The system has two distinct networks, the payload and the controlled networks (see Figure 1), both are local area networks (LANs). The former is asynchronous and is used for application data transfers. There are no assumptions based on time on the payload LAN and it is used for client-server communication. The later, used for server-server communication, is a controlled LAN composed by physical machines, where is implemented a Distributed Shared Register (DSR) [16]. The consensus service uses the DSR to execute the crucial parts of the consensus protocol. The DSR:

(payload) and, on the clients point of view, the virtual machine is transparent, meaning clients can not recognize the physical-virtual architecture. Each machine has only one network interface (NIC), a firewall and/or bridge mode are used in the host to ensure the division of the networks.

We assume that host vulnerabilities cannot be explored by the virtual machine. The virtual machine monitor (VMM) ensures this isolation, meaning the attacker has no way to access the host through the virtual machine. This is a premise in the virtualization technologies, such as VirtualBox, LVM, XEN, VMWare, VirtualPC, etc. Our model assumes that the host system is not accessible externally, which is also granted by the use of bridge mode and/or firewalls on the host system.

Distributed Shared Register (DSR): We created the DSR, an emulated shared memory [16] based on message passing over a controlled LAN and making use of local files. We assume the controlled LAN to be only accessed by components of the DSR. The DSR is implemented in the virtual machine host and we assume that the VMM ensures isolation between the host and the guest.

The DSR performs just two operations: (1) *read()*, that reads the last message written in the DSR; and (2) *write(m)* that writes the message m in the DSR. We assume two properties about these operations: (i) liveness, meaning that the operation eventually ends; and (ii) safety, i.e., the read operation always returns the last value written. To ensure this properties, in each server we created a file where the guest has write-only access and another file where it has read-only access and the access is made by a single process [16]. The first file is the server space, and no other server can write on it. The second one is the other servers register, updated by the DSR. The VMM provides the support to make a file created in the host to be accessible to the guest, enforcing the write-only/read-only permissions.

The DSR only accept typed messages, and there is only three types, (i) PROPOSE, (ii) ACCEPT and (iii) CHANGE. The untyped or mistyped messages are ignored. We assume that the communication is made by fair links with the following properties: if the client and the recipient of a message are both correct then (1) if a message is sent infinitely often to a correct receiver then it is received infinitely often by that receiver; (2) there exists some delay T such that if a message is retransmitted infinitely often to a correct receiver according to some schedule from time t_0 , then the receiver receives the message at least once before time $t_0 + T$; and (3) the message is not modified in the channel [6] [20]. This assumption appears reasonable in practice, since the DSR is running in a separated synchronous LAN and can only fail by crash, based on the VMM isolation.

3.1 Properties

The problem of atomic multicast, or total order reliable multicast, is the problem of delivering the same messages in the same order to all processes of a system. The definition in byzantine context can be seen as the following properties [7]:

Validity - If a correct process multicasts a message M , then some correct process eventually delivers M .

Agreement - If a correct process delivers a message M , then all correct processes eventually deliver M .

Integrity - For any message M , every correct process p delivers M at most once, and if $\text{sender}(M)$ is correct then M was previously multicast by $\text{sender}(M)$.

Total Order - If two correct processes deliver two messages M_1 and M_2 then both processes deliver the two messages in the same order.

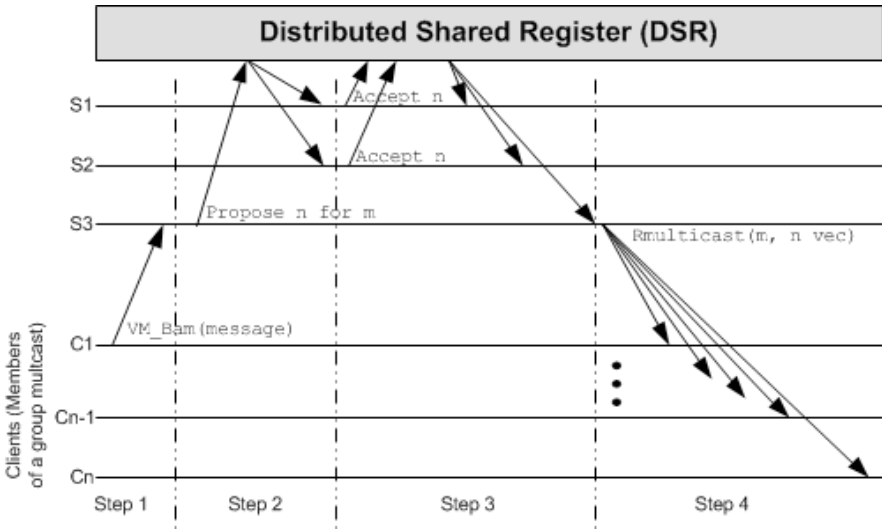


Fig. 2 Atomic multicast flow

4 Vbam Algorithm

Discussion: The process needs only one server as sequencer. This server is responsible for proposing orders for client messages. Other servers are replicas of the service. Initially, the sequencer is the process which the id is zero. The sequencers change happens every time most servers ($f + 1$) agree to be necessary. As in others byzantine fault tolerant (BFT) system [5], [6], [15], to deal with the problem of a malicious server p_j , that would discard a message. Therefore the client p_i waits for receiving its own message with an order number for a T_{resend} time. After this T_{resend} the client send its message to all servers. A correct server when receives the client message, and it is not the sequencer, asks for a sequencer change. If $f + 1$ correct servers ask for a sequencer change, then it will be performed and the protocol makes progress. However, the payload system is assumed to be asynchronous, so there are no bounds on communication delays, and it is not

possible to define an "ideal" value for T_{resend} . Correia [15], shows that the value of T_{resend} involves a tradeoff: if too high, the client can take long to have the message ordered; if too low, the client can resend the message without necessity. The value should be selected taking this tradeoff into account. If the command is resent without need, the duplicates are discarded by the system.

This section offers a deeper description of the algorithm. The sequence of operations of the algorithm is presented in the sequencer and destination nodes. It is first considered the normal case operation and, following, the execution where faults do exist. The flowchart of the normal case operation can be seen in the Figure 2. For clarity of presentation, we consider a single group multicast.

4.1 Normal Operation

- step 1) The process starts when some client c_i sends to the sequencer an ORDER message $\langle ORDER, m, t, v \rangle_{\sigma_s}$ with the message m included. The field t is the timestamp of the message content to ensure one-time order semantics, in other words, servers will not order the message if t is not bigger than $t - I$ for c_i . This politics prevents multiple ordering of a single message. The field v is vector that takes a MAC per server, each obtained with the key shared between the client and that server. Therefore, each server can test the integrity of the message by checking if its MAC is valid, and discard the message otherwise. In case the message has already been ordered, the server resends it to the client.
- step 2) After verifying that MAC in v is correct and the timestamp is valid for the client's message, the sequencer generates a PROPOSE message as $\langle PROPOSE, n, o, mac \rangle_{\sigma_p}$, where o is the original message, n the ordering number for it and mac is the message authentication code for m . The DSR automatically input in the proposal message the ID of the sequencer. The server will expect for the acceptance of the message, i.e., f processes agreeing with the proposal. Then the server, accept this order and saves it in the atomic buffer. All messages sent in the DSR will be delivered if the sender and the receiver are not crashed, as we have discussed in 4.
- step 3) By receiving a proposal, the server s_k validates it, meaning that (i) s_k verifies, using the MAC in v , if the content of the message m is correct and (ii) verifies if there is no other proposal accepted before that with the same sequence number n . Then the proposal is accepted by s_k that writes an ACCEPT message on it's reserved space of the register. The message format is $\langle ACCEPT, n, h_m, mac \rangle_{\sigma_p}$ and it contains the hash of the client's message h_m , the ordering number n to m and a message authentication code mac for m . After writing the accept message, the process waits for $f - I$ acceptance messages to save it in the atomic buffer.
- step 4) The sequencer reliably multicast the message with the order and the mac vector from at least $f + I$ different servers that accepted it. After receiving and validating the vector the clients finally accepts it and delivers in the proposed order.

4.2 Faulty Operation

The faulty operation implies that a change will happen, therefore here is a brief explanation of how it develops.

Sequencer Change: During the system configuration, all servers receive an identification number. These numbers are sequential and start at zero. All servers can recognize the identifier i of the sequencer and the total number n of servers in the system. When $f + 1$ correct servers suspect of the current sequencer they simply define $i = i + 1$ as the next sequencer if $i < n$, otherwise $i = 0$.

When validating a proposal the server s_k verifies if the message's content is correct using its MAC in the vector v and if the proposal is correct based on the earlier accepted proposals. If the message, for some of the reasons above, is not valid, then s_k will ask for a sequencer change.

- a) Correct servers can enter in the faulty operation by the two following ways:
 - i) When the server s_k receives a change message e , but does not suspect the server s_s yet, the process just stores e in its local buffer.
 - ii) When the process s_k suspects of the server s_s about a single message m , then s_k writes in the DSR and in its own buffer a new message $\langle CHANGE, sid, h_m, S_s \rangle_{\sigma_p}$ that contains sid as its own identifier, the hash of the message h_m and s_s as the id of the server for which it suspects.
- b) The server s_k starts a search in its buffer trying to find $f + 1$ change messages that relate to m and the server s_s . In case s_k finds $f + 1$ (including server s_k) different sid to the same propose message, then the server changes the sequencer to $s_s + 1$. If s_s is its own id then, based on the last accepted messages, it restarts the ordering.

With the new sequencer the protocol makes progress as in normal case operation.

5 Implementation and Evaluation

The algorithms were implemented using the programming language Java, JDK 1.6.0, and executed using JVM Sun. The communication channels were implemented using TCP sockets from API NIO. The operational systems of the virtual machine hosts used were MacOSX Lion and Ubuntu 12.04. On virtual machines themselves, we used Ubuntu 12.04, Ubuntu 12.10 and Debian 6 Stable, being VirtualBox the VMM of choice. The adopted evaluation metric is latency, since this is one of the metrics most widely used to evaluate computational systems and that it represents the efficiency of the system in a very simple way [5], [6], [7], [8].

The values were obtained through micro-benchmarks in different loads. The latency was obtained measuring the round-trip time, that is, we measured the time between sending and receiving a group of messages. The reasoning behind the use of micro-benchmarks is to properly measure the algorithm without external influences. In order to gauge the protocol's capacity, we ran it with different message sizes.

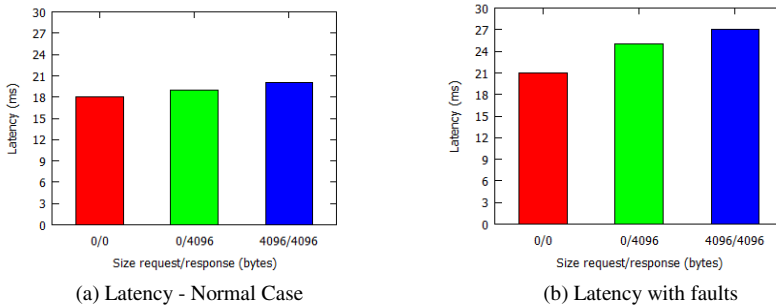


Fig. 3 Performance Evaluation

To evaluate the algorithm's performance in the absence of faults, we ran it in normal conditions and sent 10.000 messages by a single client with three different loads: 0/0 kb, 0/4 kb and 4/4 kb. With those we have: an empty message and an empty ordered message, an empty message and a 4kb ordered message and a 4kb message with an 4kb ordered message. To evaluate the algorithm in a faulty environment, we conducted the experiment with $f = 1$ faulty server. In 3(a) and 3(b) we demonstrate the latency to each different load. The latency was obtained by the average from all total order multicasts. As we can observe, the latency has minimal variations between different loads.

We show the comparative data between our approach and the state of art for atomic multicast on the table 1. All the numbers consider non-faulty execution. Our approach benefits are visible as number of communication steps and resilience since we consider systems subject to byzantine faults.

6 Conclusion

By exploring the use of the Distributed Shared Register and virtualization techniques, we have managed to propose a simple inviolable LAN that supports our BFT atomic multicast. It was showed that it is possible to implement a reliable consensus service with only $2f + 1$ servers using common technologies, as is virtualization and data sharing abstractions. The virtualization technology is widely used and can provide a good isolation between the servers and the external world, and the use of the DSR makes it easy to maintain the protocol progress.

References

- [1] Défago, X., Schiper, A., Urbán, P.: Total order broadcast and multicast algorithms: Taxonomy and survey. *ACM Computing Surveys (CSUR)* 36(4), 372–421 (2004)
- [2] Rodrigues, L., Veríssimo, P., Casimiro, A.: Using atomic broadcast to implement a posteriori agreement for clock synchronization. In: *Proceedings of 12th Symposium on Reliable Distributed Systems*, pp. 115–124. IEEE (1993)

- [3] Kemme, B., Pedone, F., Alonso, G., Schiper, A., Wiesmann, M.: Using optimistic atomic broadcast in transaction processing systems. *IEEE Transactions on Knowledge and Data Engineering* 15(4), 1018–1032 (2003)
- [4] Bessani, A.N., da Silva Fraga, J., Lung, L.C.: Bts: a byzantine fault-tolerant tuple space. In: *Proceedings of the 2006 ACM Symposium on Applied Computing, SAC 2006*, pp. 429–433. ACM, New York (2006)
- [5] Castro, M., Liskov, B.: Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)* 20(4), 398–461 (2002)
- [6] Yin, J., Martin, J., Venkataramani, A., Alvisi, L., Dahlin, M.: Separating agreement from execution for byzantine fault tolerant services. *ACM SIGOPS Operating Systems Review* 37(5), 253–267 (2003)
- [7] Correia, M., Neves, N., Veríssimo, P.: From consensus to atomic broadcast: Time-free byzantine-resistant protocols without signatures. *The Computer Journal* 49(1), 82–96 (2006)
- [8] Favarim, F., Fraga, J., Lung, L.C., Correia, M., Santos, J.: Exploiting tuple spaces to provide fault-tolerant scheduling on computational grids. In: *10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, ISORC 2007*, pp. 403–411 (May 2007)
- [9] Ekwall, R., Schiper, A., Urbán, P.: Token-based atomic broadcast using unreliable failure detectors. In: *Proceedings of the 23rd IEEE International Symposium on Reliable Distributed Systems*, pp. 52–65. IEEE (2004)
- [10] Reiter, M.: Secure agreement protocols: Reliable and atomic group multicast in rampart. In: *Proceedings of the 2nd ACM Conference on Computer and Communications Security*, pp. 68–80. ACM (1994)
- [11] Guerraoui, R., Schiper, A.: The generic consensus service. *IEEE Transactions on Software Engineering* 27(1), 29–41 (2001)
- [12] Pieri, G., da Silva Fraga, J., Lung, L.: Consensus service to solve agreement problems. In: *2010 IEEE 16th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 267–274. IEEE (2010)
- [13] Correia, M., Veríssimo, P., Neves, N.: The design of a cots real-time distributed security kernel. *Dependable Computing EDCC-4*, 634–638 (2002)
- [14] Veríssimo, P.: Travelling through wormholes: a new look at distributed systems models. *ACM SIGACT News* 37(1), 66–81 (2006)
- [15] Correia, M., Neves, N., Veríssimo, P.: How to tolerate half less one byzantine nodes in practical distributed systems. In: *Proceedings of the 23rd IEEE International Symposium on Reliable Distributed Systems*, pp. 174–183. IEEE (2004)
- [16] Guerraoui, R., Rodrigues, L.: *Introduction to reliable distributed programming*. Springer-Verlag New York Inc. (2006)
- [17] Lamport, L., Shostak, R., Pease, M.: The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4(3), 382–401 (1982)
- [18] Rodrigues, R., Castro, M., Liskov, B.: Base: Using abstraction to improve fault tolerance. *ACM SIGOPS Operating Systems Review* 35(5), 15–28 (2001)
- [19] Menezes, A., Van Oorschot, P., Vanstone, S.: *Handbook of applied cryptography*. CRC (1996)
- [20] Wangham, M.S., Lung, L.C., Westphall, C.M., Fraga, J.: Integrating SSL to the JACOWEB security framework: Project and Implementation. In: *IM 2001*, pp. 779–792 (2001)

An Approach to Automated Verification of Multi-Level Security System Models

Andrzej Stasiak and Zbigniew Zieliński

Military University of Technology, Warsaw, Poland
{astasiak, zzielinski}@wat.edu.pl

Abstract. In the paper the approach to the multi-level security (MLS) system models verification is presented. In the work the MlML profile was developed with possibility of the confidentiality or integrity verification on the base of Bell-LaPadula or Biba models. The Bell-LaPadula and Biba models are formalized together with scenarios that represent possible run-time instances. Properties of the security policy model are expressed as constraints in OCL language. The feasibility of the proposed approach by applying it to a non-trivial example is demonstrated.

1 Introduction

The issue of building a reliable Specialized Computer Systems (SCS), that process data with different levels of sensitivity becomes particularly relevant to governmental, military or financial institutions. The problem of processing information with different levels of sensitivity has been extensively studied since the early 70s of the twentieth century [1–3]. Various multilevel security models (MLS) have been created to enforce confidentiality and integrity of data. Some of the more popular models are Bell-LaPadula (BLP) Model [2–3], Biba Model [4], Lipner’s Integrity Matrix Model and Clark-Wilson Model [5].

In a system development process, the security is usually considered as a nonfunctional requirement, but unlike other nonfunctional requirements, such as reliability and performance, security has not been fully integrated within the development lifecycle and it is still mainly considered after the design of the system [6]. On the other side, in the MLS systems security requirements introduce not only quality characteristics but also constraints under which the system must operate (see §4.). Ignoring such constraints during the development process could lead to serious system vulnerabilities.

The basic idea of integrating system design models (expressed in UML) with security considerations is not new [6-12]. Such integrated models with both a concrete notation and abstract syntax are called security-design models [8]. We have extended this notion on to the MLS security-design models. In [7] we have proposed a method of software design of MLS-type systems called MDmls, which is based on MDD (Model Driven Development) approach [13]. The essence of the MDmls method is the integration of MLS security models with system design models expressed in UML-based language.

The idea of formulating OCL queries on access control policies was introduced in [10-12], who first explored the use of OCL for querying RBAC policies. In the work [8] the security modeling language, called SecureUML, was presented, which is closely related to Role Based Access Control (RBAC). Subsequently, in [9] was shown that security properties of security-design models could be expressed as formulas in OCL and an expressive language for formalizing queries concerning RBAC policies was also proposed. However, RBAC has several limitations and its use as a base for modeling security policy of MLS systems is impractical.

We see our contributions as follows. Firstly, we have extended the approach proposed in [8-9] to the MLS security models, which are based on the Bell-LaPadula and Biba models. Secondly, we demonstrate, how to develop special UML profile (meta-model) for MLS systems (called MIsML) and how to formulate OCL expressions to check properties of the MLS models. Next, we show the feasibility of this approach by applying it to a non-trivial example: MLS security policy and security-design models verification of the Secure Workstation for Special Application (SWSA) Project¹ with the use of IBM RSA tool.

The rest of the work is organized as follows. In Section 2 we describe our general approach to security modeling in MLS systems. In Section 3 we propose MIsML profile for the confidentiality and integrity verification with BLP and Biba models accordingly. In Section 4 we describe an example of MLS security policy and MLS security-design models verification. In Section 5 we draw conclusions and discuss future work.

2 General Approach to Security Modeling in MLS Systems

In this section we explain our approach to analyze properties of security-design models of MLS systems on the base of the BLP or Biba models and the use of the evaluation and simulation of the models.

2.1 Problem Statement

Secure software design of the MLS systems employs dedicated tools to verify the confidentiality and the integrity of data using UML models. In general, the UML security models could be embedded in and simulated with the system architecture models, thus the security problems in MLS system can be detected early during the software design.

As it was shown in [9], precise analysis of UML models (or their instances) depicted by some diagrams requires definition of the formal semantics of diagrams, that is, definition of an interpretation function $I(\cdot)$, which associates formal (mathematical) structures $I(M)$ to well-formed diagrams M . In general, given

¹ Project No. OR00014011 supported by The Polish National Center for Research and Development.

a security modeling language with a formal semantics, one can reason about models by reasoning about their semantics. In the case of a MLS security modeling language \mathcal{L}_{MLS} , a security model (or a security model instance) M has a property P (expressed as a formula in some logical language) if and only if $M \Rightarrow P$.

We formulate the problem as follows: because OCL is the natural choice for querying UML models, it can be used to constrain and query MLS security-design models. Our approach to analyze properties of MLS security-design models and their instances reduces deduction to evaluation and simulation.

The way of formally analyzing MLS security policies concerning confidentiality and integrity of a designed system and modeled by model M we obtain by expressing desired properties as OCL queries and evaluate these queries on the UML models or model instances.

2.2 *MLS Models*

The Bell-LaPadula model focuses on data confidentiality and controlled access to classified information. In this formal MLS model, the entities in an information system are divided into subjects and objects, all subjects and objects are labeled with a security level. The levels represent the relative sensitivity of the data and the clearance of the user on whose behalf the subjects are operating. For semantic reasons of model building the security level of subjects and objects will be distinguished.

Let $C = \{c_1, c_2, \dots, c_L\}$ denote the ordered set of clauses which represent sensitivity of data used in the MLS system, where $c_i \leq c_{i+1}$ for $1 \leq i < L$. Let $IC = \{\theta_1, \theta_2, \dots, \theta_C\}$ be the set of categories of information processed in the system.

For each subject $s \in S$ we assign a security level $SL(s)$ as a pair $\langle c, A \rangle$ and for each of object $o \in O$ we assign a security context $SC(o)$ as a pair $\langle c', A' \rangle$, where $c, c' \in C$ and $A, A' \subseteq IC$. Security levels and security contexts can be compared. It could be noticed that not all pairs of levels are comparable. This leads to the use of the concept of lattice of security levels.

A dominance relationship $dom(s, o)$ may be introduced between subject $s \in S$ with $SL(s) = \langle c, I \rangle$ and object $o \in O$ with $SC(o) = \langle c', I' \rangle$, if $SL(s) \geq SC(o)$. It can be expressed as the formula:

$$dom(s, o) \Leftrightarrow (c \geq c') \wedge (I' \subseteq I). \quad (1)$$

The BLP model is based around two main rules: the simple security property and the star property [3]. The simple security property (ss-property) states that a subject $s \in S$ can read an object $o \in O$ if the formula (1) is hold. The simple security property prevents subjects from reading more privileged data. The star property (*-property) states that a subject can write to an object, if subject is dominated by object.

The Bell La-Padula model does not deal with the integrity of data. It is possible for a lower level subject to write to a higher classified object. The Biba model

addresses the problem with the star property of the Bell-LaPadula model, which does not restrict a subject from writing to a more trusted object. Similarly to BLP model integrity levels are defined by labels, consisting of two parts: a classification and a set of categories. Each integrity level will be represented as $IL = \langle \alpha, A \rangle$ where IL is the integrity level, α is the classification and A is the subset of categories. Then, similarly to BLP model, the integrity levels then form a dominance relationship. The Biba model is actually a family of different policies that can be used depending on the specifics of MLS system [4].

2.3 *Metamodel*

In our approach, in order to determine the system's security of MLS type, we refer to the UML four-layer hierarchy [8,14]. In order to simplify the process of constructing a specialized MIsML language, the specification will be determined at the level of M2, which, in essence, is the same metamodel. In the metamodel the concepts will be described that are relevant to the domain, and which will be mapped to the UML profile stereotypes with developed MIsML language. In this sense, our proposal of MIsML is an extension of the UML language, not its complete specification.

Level M1 would be a model of the field (description of the MLS system security), in which we use elements developed from meta-model, for the construction of MIsML language expressions in the form of diagrams. Each diagram would be created according to the rules of the UML language, enhanced with a new profile property (which is why it is acceptable to refer to the developed diagrams of the field, both for the concepts of language specified in the profile, as well as in the UML language).

2.4 *Approach*

Due to the significant practical limitations of the use of RBAC policies for constructing specialized systems of MLS type, (inspired by the work [14] have proposed our own language (meta)model MIsML and achieved formalization of restrictions for the specific BLP and Biba models). Our proposal relates mainly to possible formalization of restrictions in the OCL language, towards the relationship between the users (subjects), facilities, privileges, performed actions and certain states of the system. Contrary to the work [14], which also proposes its own tool that implements proposed methodology, we have based our solution on a typical CASE environment that has appropriate support for the UML models validation process (actually DSL) in the "starting" defined OCL limitations and UML models (expanded by the semantic action language). Our approach has been validated in the IBM RSA tool with Simulation Toolkit.

Our approach to analysis of the model properties of MLS security-design models and their instances leads to the evaluation and simulation. The integration

of security models with models of systems described in UML enables the simulation, which allows verifying the security properties of the designed system MLS software or the security policy models at the stage of analysis and modeling.

3 The Profile MIsML

The MIsML profile proposed in the article was developed as an extension of previously proposed solution [7], [14] with additional possibility of the integrity verification on the base of the Biba model.

The creation of a profile precedes the detailed analysis of the areas for which it is developed [15]. The result of this analysis in step 1 is a glossary of terms of the field, which in our project consists of: *Subject*, *Object*, *Action Requested*, *Permission*, *Security Context*, *Security Level*, *Action*, and *Information Category* (the last three are the properties of concepts). In the next step (Step 2) verbal definitions are transformed into the domain class model. In the final step (Step 3) the model of domain is analyzed, and then its elements are mapped to stereotypes, classes and properties, which leads us to UML metamodel (M2 level).

The process of the MIsML profile construction was performed independently for description of discretionary access control to resources (security access profile) and mandatory access control based on the base of BLP and Biba models and it was portrayed in detail in the following two subsections.

3.1 The Metamodel of Discretionary Access Control

The profile building process begins with a creation of metamodel. Type of metamodel of discretionary security protection for MIsML profile is shown in Fig. 1, in which five stereotypes (concepts), including three classes were described: *Subject*, *Object*, *Permission*, *Association Class: RequestedAction*, and their property: *Action*. Within the profile several constrains were defined. Now we give some examples of them.

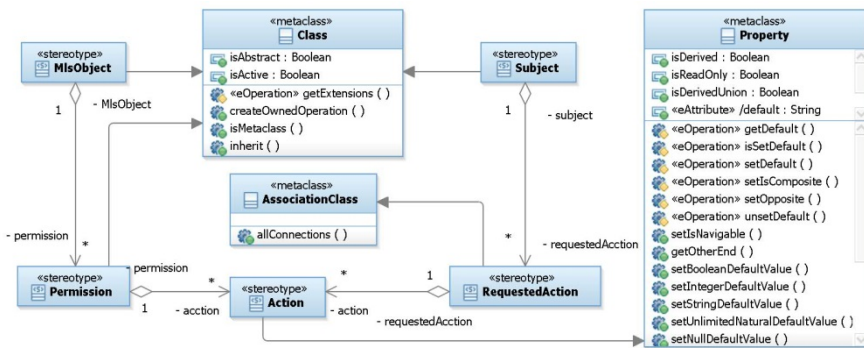


Fig. 1 Security profile (DAC – model)

It is required that the specific access request of the subject to the object would refer to existing entities and objects: if there is a connection in the request with the object, then there is also a link to the subject and if there is a connection in the request with the subject, then there is also a link to the object.

context RequestedAction **inv:**
 ((self.base_AssociationClass.getAppliedStereotype('MlsML::MlsObject')<>null) **implies**
 ((self.base_AssociationClass.getAppliedStereotype('MlsML::MlsObject')<>null) **and**
 (self.base_AssociationClass.getAppliedStereotype('MlsML::Subject')<>null))

3.2 *MLS Metamodel*

Metamodel of security type BLP for MlsML profile is shown in (Fig. 2). In this metamodel four stereotypes were defined, including: two classes that describe the *Security Context*, *Security Level* in the BLP model and two of their properties, i.e. *Clause* and *Information Category*, and also written below (in OCL language) required restrictions as follows.

It is required that for the BLP model there were two mandatory defined properties: *Clause* & *InformationCategory*; for the Biba model there was mandatory defined property: *LevelOfIntegrity*, (those properties should appear separately, i.e., in the diagram we do not combine BLP model with the Biba model). It should be noticed that following restriction should be applied as well in the context of *SecurityLevel*.

context SecurityContext **inv:**
if self.getAllAttributes().getAppliedStereotypes() -> size() = 2 **then**
 ((self.getAllAttributes().getAppliedStereotypes().name -> count('Clause') = 1) **and**
 (self.getAllAttributes().getAppliedStereotypes().name -> count('InformationCategory') = 1))
else if self.getAllAttributes().getAppliedStereotypes() -> size() = 1 **then**
 self.getAllAttributes().getAppliedStereotypes().name -> count('LevelOfIntegrity') = 1
else false endif endif

It is required that the integrity level only exists as an attribute of the context or security level. For the BLP model was defined two mandatory properties: *Clause* & *InformationCategory*.

context LevelOfIntegrity **inv:**
 (self.base_Property.owner.getAppliedStereotype('MlsML::SecurityLevel')<>null) **or**
 (self.base_Property.owner.getAppliedStereotype('MlsML::SecurityContext')<>null)

The OCL constraints, defined above, could be verified in three modes:

- debug: in which the OCL expression can be activated in a specific context, with the possibility of tracking the results of its execution;
- as a validation in batch mode: then the starting of defined constraints is directly achieved by selecting the "validate" function and applies only to the specified element (which can also be the whole diagram);

- as a validation in the live mode: that is activated at the moment of insertion of a new element to a diagram, and directly affects only this element (or as well the elements directly linked to it).

Such validation in the live mode, in our opinion, is the most interesting tool that allows the security officials to create only the correct security policies, i.e., each insertion of a new item of the policy is validated against predefined rules.

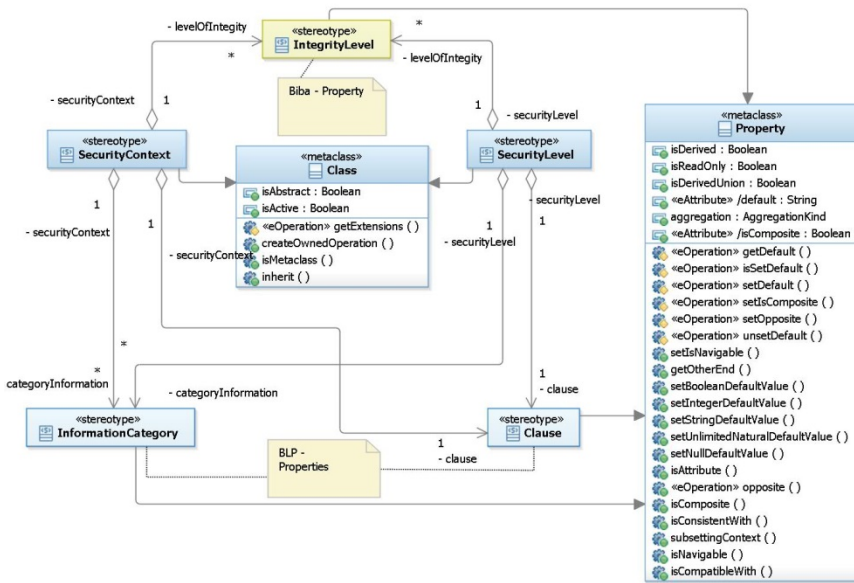


Fig. 2 Security profile (MAC – BLP & Biba models)

3.3 Using the MlML Profile

The profile application we begin from building the model of M1 level. In our project, the key aspect dominating its success was the creation of dominance function (`checkDom()`) and verification its correctness of accomplishing the specific action requested on the object (in our example - virtual machine image - VMI) by specified subject (user) (`checkDAC()`). The constrains in OCL language for this functions are presented below.

```

context User2VMI_RequestedAction::checkDAC():Boolean
pre: self.Request -> notEmpty() and self.vmi.OID <> null and self.user.UID <> null
self.user.permissionU2V.UID = self.vmi.permissionU2V.UID and
self.user.permissionU2V.OID = self.vmi.permissionU2V.OID
post: result = let oi : Integer = self.vmi.OID in let us : Integer = self.user.UID in
self.user.permissionU2V->select(per:PermissionU2V|(per.OID= oi) and (per.UID=
us)).Permissions -> includes(self.Request)
    
```

```

context User2VMI_RequestedAction::checkDom():Boolean
pre: (self.vmi.vMI_SecurityContext -> notEmpty()) and (self.user.user_SecurityLevel ->
notEmpty())
post: result = (self.user.user_SecurityLevel.SLInfCat -> include-
sAll(self.vmi.vmi_VMI_SecurityContext.SCInfCat)) and
(if self.user.user_SecurityLevel.SLClause >= vmi.vmi_VMI_SecurityContext.SCClause then true
else false endif)
    
```

It was assumed that in the implementation of the above constraints the attribute *Clause* from the set {*Unclassified, Restricted, Confidential, Secret, TopSecret*} takes (accordingly) integer values from the set {0, 1, 2, 3, 4}. The same profile may be used for the integrity verification, but in the current implementation (because of specific of SWSA project) the models based on BLP and Biba are treated as disjoint. In the proposed solution, the profiles for BLP (Fig. 3) and Biba models differ by the attributes of classes with stereotypes *SecurityContext* and *SecurityLevel*, and by methods of the association class *RequestedAction*. In the place of the BLP model class attributes (Fig. 3) such as *InformationCategory* and *Clause* - in the Biba model the attribute *Integrity Level* was specified and in the place of the *checkDom()* method for the BLP model – in the Biba model the *checkIntegrity()* method appears.

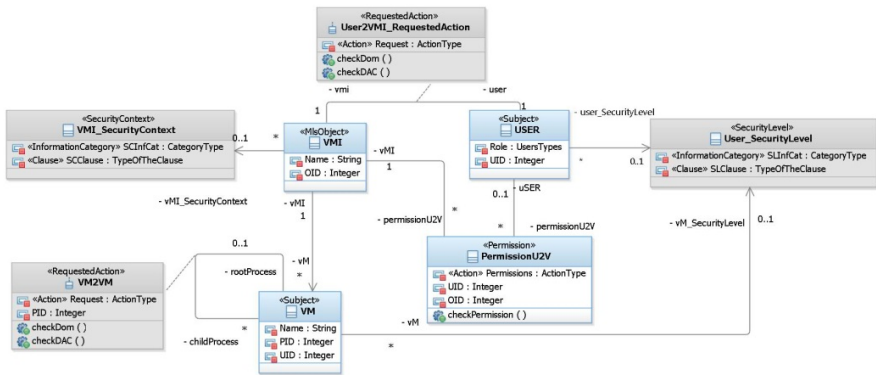


Fig. 3 M1 level for MisML metamodel (BLP access control model)

4 Case Study: A Security Policy Model Verification and Simulation

In the following part of work, we will present an example to illustrate the use of the proposed method for the construction of the MLS security policy used in the SWSA project. The domination relationship is shown in the model as the operation of the U2VMI association class (Fig. 3) and in OCL language it was implemented as constraints that this operation must fulfill. We define constraints for a created model of security policy to run virtual machines by checking the domination relationship.

4.1 *Secure Workstation for Special Application(SWSA) Project*

In SWSA project to ensure the security of multilevel classified data an approach to develop software on the base of the virtualization technology for the separation of independent security domains was taken. To develop this type of MLS system the integration of available virtualization technology (software and hardware), application of formal methods for both ensuring and control of the confidentiality and integrity of data are needed. A natural way to build such systems is component approach, which assumes the use of ready and available hardware components and software, in particular virtualization packages (hypervisors) available as open source like Xen.

Developed within SWSA project software should allow for the simultaneous launch of several specific instances of operating systems on one PC (such as a workstation or server) designed to process data of different classification levels (e.g., public and proprietary), or to process the data in different systems, for which there is need for separation of data.

The key element of SWSA is Virtual Machines Monitor (VMM), which is responsible for control of running virtual machines in accordance with defined MLS security policy and their switching to ensure the separation of resources. It was assumed that the proposed VMM software should make it possible to simultaneously launch several (of many possible) instances of special versions of operating systems on a single computer with the provision of: access control, separation of resources, cryptographic protection, and strict control of data flow.

In the SWSA environment one can distinguish three types of actors: the system administrator, the security officer and user. The security officer with the administrator and others are developing special security requirements of the system (the MLS security policy), and safe operation procedures.

4.2 *SWSA Security Policy Verification – An Example*

Example 1. The security policy specification for SWSA.

Subjects (Users) = {*DC, CSO, OD, SEC*}, where *DC, CSO, OD, SEC* denotes the SWSA users, accordingly, Deputy Chief, Commanding System Officer, Operation Director and Secretary.

Objects (Virtual machines) = {*VM11, VM12, VM13*} where *VMi*, $i \in \{1..3\}$ denotes images of virtual machines *VMi*.

Let the set of numbers *InfCat* = {1..6} represent set of information category as follows 1- Infrastructure, 2 – Strategy and Defense, 3 – Personnel, 4 – International Security, 5 – Commanding Systems, 6 – New Weapons.

In the system SWSA the ordered set *C* of clauses is used (from the lowest to highest): $C = \{Unclassified(Uncl), Restricted(Res), Confidential(Conf), Secret(Sec), Top Secret(TSec)\}$.

The MLS security policy claims security level $SL(s) = (c, Inf c)$ assigned to each of subject $s \in Subjects$ and security context $SC(o) = (c', Inf c')$ assigned to each of object $o \in Objects$, where $c, c' \in C$ and $Inf c, Inf c' \subseteq Inf Cat$.

$SL(DC) = (Sec, \{1..6\})$, $SL(OD) = (Sec, \{3,4,5,6\})$, $SL(CSO) = (Sec, \{2,5,6\})$, $SL(SEC) = (Conf, \{3,4,5,6\})$, $SC(VM1) = (Sec, \{1,2,4\})$, $SC(VM2) = (Conf, \{3,4\})$, $SC(VM3) = (Sec, \{5,6\})$.

Assume that one of the actions from the set $Actions = \{Run, Modify\}$ may be requested by user, which may concern the selected virtual machine and users have assigned permissions to virtual machines from the set $Permissions = \{Read (R), Read Attr (RA), Write (W)\}$. Let $Run(u, v)$ be a predicate that user u is allowed to take the action Run on a virtual machine v . Thus the following rules are holding $Run(u, v) \Rightarrow Perm(u, v) - \{R, RA\} \neq \emptyset$, $Modify(u, v) \Rightarrow Perm(u, v) - \{R, RA, W\} \neq \emptyset$, where $Perm(u, v)$ is the subset of permissions assigned the user u to the virtual machine v .

Assuming that all users have granted permissions R, RA to all *Virtual Machines* we easily verify MLS security policy given in the example e.g. we prove that $Run(DC, VM1) = true$, $Run(OD, VM2) = false$ etc.

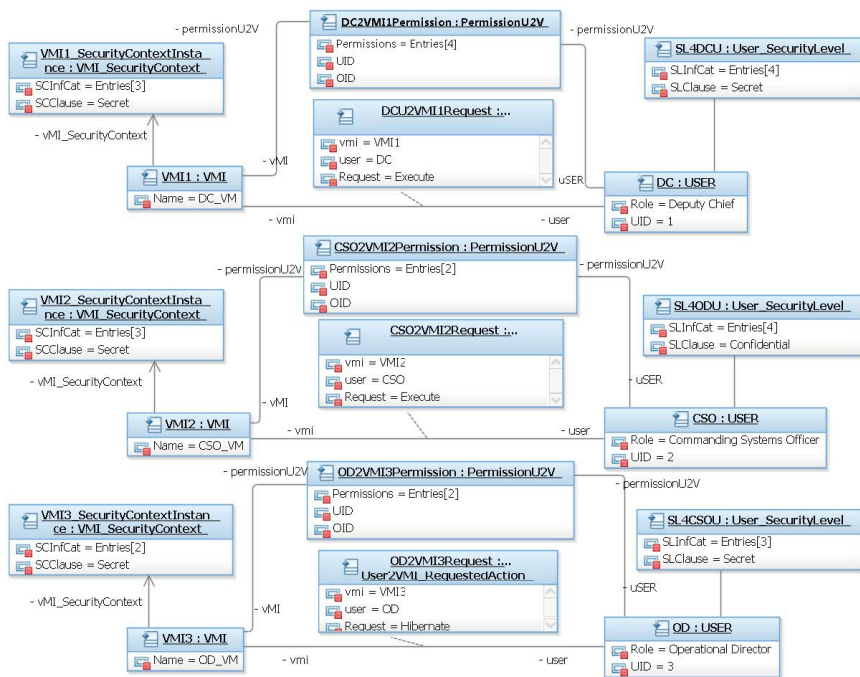


Fig. 4 Case study – implementation of the decision of Example 1 in special tool for verification MlsModel

The presented model of M0 level (Fig. 4) is an instance of decision of security officer in the security policy matter, described in the Example 1.

The constructed policy is analyzed up to date (in the process of its construction, i.e., each insertion of a new item is preceded by the checks of constraints) and subjected to a live and batch verification, enabling the correction of committed errors resulting from non-compliance of the pre-defined rules (defined by meta-model and model levels).

4.3 Analysis of an Example Scenario

The process of running of the virtual machine (Fig. 5) on the base of the virtual machine image $vmi_j, j \in \{1, 2, \dots, K\}$ with defined security context $sc(vmi_j)$, by the subject (user) u_i with defined security level sl_i we will describe as $u_i \xrightarrow{RUN} vmi_j$. The process running of the virtual machine (subject) will be generated on the basis of the image vm_{j_k} , and its current security level (csl) for a subject vm_{j_k} will be defined as a pair $\langle clause, infCat \rangle$ from the following dependency:

$$csl := \langle \min \left(Clause(sl(u_i)), Clause(sc(vmi_j)) \right), InfCat(sl(u_i)) \cap InfCat(sc(vmi_j)) \rangle,$$

where $InfCat(e)$ is the set of information categories of the element $e \in U \cup VMI$.

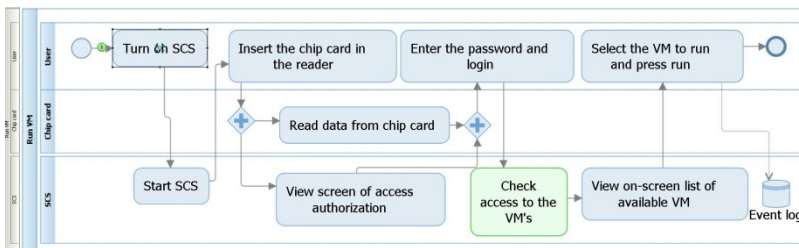


Fig. 5 The business model of a process of virtual machine running in the SWSA system

For verification of the models behavior we propose the use of the simulation mechanisms of UML models with the semantics action language as an extension. This is particularly important because the OCL language does not allow us to express constrains based on the states of models (there can be no changes in the characteristics of class instances (objects)). The diagram presented in Fig. 6 is an example of diagram of the course of the model states simulation session of the one instance of virtual machine running in the SWSA environment.

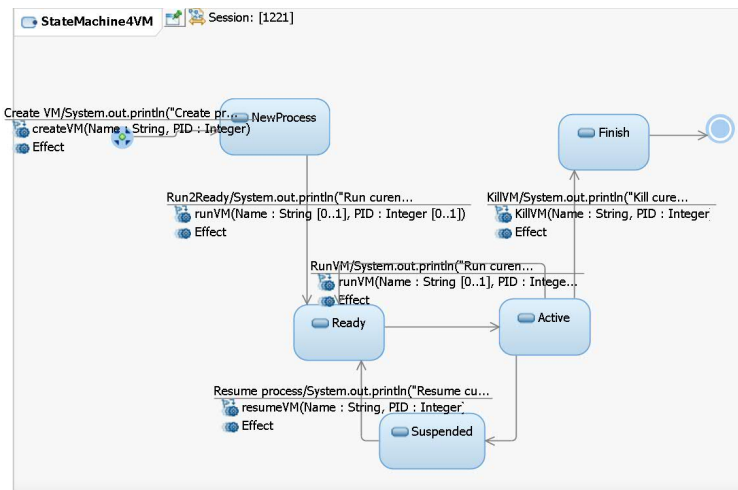


Fig. 6 The states verification of the process of running virtual machines by the model simulation

The MLS security-design model simulation facilitates an understanding of the dynamic aspects of the system and allows us to detect design errors early without necessity of real products and real environment building. The simulation model building involves constructing of the security models in the UML language, their computer implementation in the simulation environment (with UML, ALF(UAL) and OCL) and developing plan for an experiment;

The environment used in the work (IBM Rational Software Architect 8.5 with Simulation Toolkit [16]) enables you to collect the simulation results in the following forms: history of messages sent between objects, traces of messages passing control flow, history of console records. It should be noted that capabilities of this environment may be extended with the use of UAL language.

In the paper we present only two selected examples concerning simulation of the security mechanisms behavior (due to limited volume of the paper), which incorporates business model of a process of virtual machine running in the SWSA system (Fig. 5) and the diagram of the course of the model states simulation session of the one virtual machine instance running in the SWSA environment (Fig. 6).

By simulation, these hypotheses are verified, which could not be verified by constraints, which depends on the system state. They may involve, for example, the dynamics of the system states changes and system properties (Fig. 6), which could be traced in the process of simulation. For the present case study, is to test a breach security rules in the process of starting the next virtual machine (Fig. 5 and Fig. 6), or the verification of the conditions of separation of hardware resources that are allocated to different security domains.

5 Summary

One of the significant results obtained in this work is the MIsML profile, which was developed for the security policy verification in MLS type systems and incorporates discretionary and mandatory access control on the base of properties of the Bell-LaPadula and Biba models. The use of the MIsML profile enables verification of confidentiality and integrity of the designed MLS security policy and security-design models.

For verification of the models behavior we proposed the use of the simulation mechanisms of UML models with the semantics action language as an extension.

We examined the complete environment based on the IBM RSA tool for MLS security policy testing, which can be easily used by security officers. The approach proposed in the paper is also intended for software developers of MLS type systems, for the verification of the security-design models and algorithms implemented in designed software. The usefulness of this approach was confirmed in the completed SWSA project [18].

There is a number of promising directions for future work. One of them is to extend our the MIsML profile in such a way that it should enable verification of confidentiality and integrity of modeled MLS system together.

References

- [1] Anderson, J.P.: Computer Security Technology Planning Study, vol. II ESD-TR-73-51. Electronic System Division. Air Force System Command. Hanscom Field, Bedford, MA, 01730 (1973)
- [2] Bell D.E., La Padula, L.J.: Secure Computer System: Unified Exposition and Multics Interpretation, ESD-TR-75-306. ESD/AFSC, Hanscom AFB, Bedford, MA (1976), <http://csrc.nist.gov/publications/history/bell76.pdf> (accessed June 24, 2012)
- [3] Bell, D.E.: Looking Back at the Bell-La Padula Model, Reston VA, 20191 (2005)
- [4] Biba, K.J.: Integrity Consideration for Secure Computer System, Report MTR-3153 (1975)
- [5] Clark, D., Wilson, D.R.: A Comparison of Commercial and Military Computer Security Policies. In: Proc. IEEE Symposium on Research in Security and Privacy, pp. 184–194 (1987)
- [6] Mouratidis, H., Giorgini, P., Manson, G.: When security meets software engineering: a case of modeling secure information systems. *Information Systems* 30, 609–629 (2005)
- [7] Zieliński, Z., Stasiak, A., Dąbrowski, W.: A Model Driven Method for Multilevel Security Systems Design. *Przegląd Elektrotechniczny (Electrical Review)* (2), 120–125 (2012)
- [8] Basin, D., Clavel, M., Doser, J., Loddersted, T.: Model Driven Security: From UML Models to Access Control Infrastructures 15(1), 39–91 (2006)
- [9] Basin, D., Clavel, M., Doser, J., Egea, M.: Automated analysis of security-design models. *Information and Software Technology* 51, 815–831 (2009)

- [10] Ahn, G.J., Shin, M.E.: Role-based authorization constraints specification using object constraint language. In: WETICE 2001: Proceedings of the 10th IEEE International Workshops on Enabling Technologies. IEEE Computer Society, Washington, DC (2001)
- [11] Sohr, K., Ahn, G.J., Gogolla, M., Migge, L.: Specification and validation of authorisation constraints using UML and OCL. In: De Capitani di Vimercati, S., Syverson, P., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 64–79. Springer, Heidelberg (2005)
- [12] Jürjens, J.: UMLsec: Extending UML for secure systems development. In: Jézéquel, J.-M., Hussmann, H., Cook, S. (eds.) UML 2002. LNCS, vol. 2460, pp. 412–425. Springer, Heidelberg (2002)
- [13] Frankel, D.S.: Model Driven Architecture: Applying MDA to Enterprise Computing. John Wiley & Sons (2003)
- [14] Zieliński, Z., Furtak, J., Chudzikiewicz, J., Stasiak, A., Brudka, M.: Secured Workstation to Process the Data of Different Classification Levels. *Journal of Telecommunications and Information Technology* (3), 5–12 (2012)
- [15] Kelly, S., Tolvanen, J.P.: Domain-Specific Modeling: Enabling Full Code Generation. Wiley, NJ (2008)
- [16] Mohlin, M.: Model Simulation in Rational Software Architect: Simulating UML Models. IBM (2010)
- [17] Anders, E.: Model Simulation in Rational Software Architect: Activity Simulation. IBM (2010)
- [18] Kozakiewicz, A., Felkner, A., Furtak, J., Zieliński, Z., Brudka, M., Małowidzki, M.: Secure Workstation for Special Applications. In: Lee, C., Seigneur, J.-M., Park, J.J., Wagner, R.R. (eds.) STA 2011 Workshops. CCIS, vol. 187, pp. 174–181. Springer, Heidelberg (2011)

A Web Service-Based Platform for Distributed Web Applications Integration

Paweł Stelmach and Łukasz Falas

Institute of Computer Science, Wrocław University of Technology,
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
{pawel.stelmach, lukasz.falas}@pwr.wroc.pl

Abstract. Web applications built according to the MVC pattern are growing in popularity. At the same time web services: either SOAP or REST-based are also gaining momentum, but their adoption in web applications is usually limited to a smaller scope and with little management of their behaviour and state. The approach presented in this work does not enforce cloud-based solutions; however, it is highly advised to refer to web services as not only web interfaces to other monolithic web applications but try to perceive them as preferably independent, stateless software components, easily scalable and deployable in the cloud. This potentially could lead to dramatic reduction of the cost of software services with simultaneous increase of performance on-demand. The presented platform confronts the discrepancies between MVC pattern and service adoption in web applications, and is a proposition that increases performance of both service-based applications as well as their design methods. It introduces mechanisms for increasing the scope of service integration and also web application integration, at the same time making it manageable through specialized tools.

1 Introduction

Modern web applications are facing many new challenges. In the past they were static websites focused on information provision and distribution among various user groups. Nowadays advances in technology have caused a shift in both dynamic nature of the web content and means of its utilization. Now users have a plethora of electronic devices that allow them to access Internet at any time and from any location. Facing new possibilities, users often also decide to use web based productivity tools and entertainment applications instead of standard desktop applications. This especially has motivated them to work on a number of different devices all over the world without the need to install any additional software. Not only the software comes closer to the user, wherever he is, but a typical user decides to share his resources and personal data with that software, showing great trust in a cloud – a concept still vaguely understood among general public. User expectations will increase with time and even now we can observe an increase in popularity of solutions that include interoperability between different

applications. In the end, all a user needs to access all the necessary tools and resources is some kind of terminal with a web browser and it doesn't even have to be his device.

In the face of this new requirements developers try to make their applications more interconnected by introducing various integration mechanisms. Some of them involve using Java Remote Method Invocation; however, this approach is technology-specific and enforces strong dependencies between web applications. A more modern approach involves the usage of services based on WS-* and REST standards, which are flexible and extensible, yet still involve direct connections that had to be planned ahead of the development process. Other approaches like Enterprise Integration Patterns and Service Oriented Architecture offer more flexibility using web services but, again, require a rigid compatibility with the integrating architecture, which in many cases can be achieved only if the applications were created by developers from same organization.

Propositions of solutions to these problems can be found in many works across the recent decade, especially those regarding Web 2.0, Service Oriented Architecture and the Semantic Web. Web 2.0 is being described as a platform spanning all connected devices, it enables continuous software updates and consuming and remixing data from various sources ([10]). On the basis of this approach researchers often find Service Oriented Architecture ([8]) and Semantic Web ([7]) as a way to realize this vision. One of early propositions included service mashups ([6]) as a way to integrate, combine and utilize existing web resources and web APIs to deliver modern web applications to end-users. Other techniques focused more on the backend integration, proposing an integration platform for service management ([5]). An important extension to above solutions from a science standpoint is provided with service composition frameworks based on SOA architecture described in our earlier works ([9], [11], [12]), allowing for greater flexibility with direct connections to composition services, which provide executable services dynamically.

Alternatively, currently developed commercial products propose a different approach, which involves solutions that introduce themselves as Web Operating Systems (WebOS). The concept is not fully grounded and its loose interpretation leads both to application integration solutions as well as experiences similar to standard operating systems accessible via web. Most interpretations lean to the latter providing a remote-desktop-like experience with various web applications sharing resources similarly to the typical desktop operating system. The representatives of these solutions that are worth mentioning are eyeOS ([1]), GideOS ([2]) and CloudOS ([3]). The main drawback of these solutions is the fact that they are written as heavy client side script applications or are developed as web browser plug-ins. Also, applications working on such systems often have to be dedicated solutions, so they cannot exist separately on the web. Producers aim to transfer standard desktop operating system experience to the web. However, it is important to keep in mind that majority of people use web applications instead of desktop

applications, because they prefer the web patterns of usability to standard desktop usability patterns.

Gravitating towards more web-applications-centric solutions the Chrome Web Browser presents a different approach and tries to introduce the Web Browser as an "Operating System", which can be personalized by the user and in which he can install his applications bought in the Chrome Web Store. This solution allows users to share their Google identity through installed applications and view all installed applications on a dashboard displayed by the Chrome Web Browser. Also, the Chrome Web Browser introduces the "intent" ([4]) mechanism that allows application interoperability through the web. However, the main disadvantage of this solution is the fact that it is only browser specific. Related to this approach is the Google Drive web application and resource sharing mechanisms it provides to various applications. This approach is browser-independent and promises interesting results but it is also a closed solution focused on application integration via resources and not services.

The last representatives of software that has promised to deliver some of the desired functionality are integration platforms like Dell's Boomi ([13]), Snaplogic ([14]), Jitterbit ([15]) or CloudHub ([16]). All of them allow for defining the data flow with a workflow and for this purpose use specialized graphical user interfaces. All focus on delivering many already implemented mediators/plugins/snaps for integrations with services like twitter, dropbox or even from the enterprise surrounding: SAP or Oracle E-Business Suite. Most offer parameter mapping capabilities and web service execution. They reach to satisfy a plethora of web standards: ftp, ldap, jms, smb, smtp, SOAP etc. They usually provide workflow execution software and often integration servers that host the execution engines and parameters mapping services but the main applications – especially the workflow/integration designer – are usually desktop applications. Though those solutions are advertised as integration platforms they focus on integration of data provided by various applications and do not integrate the web applications themselves (so that they would share functionality – not only data). They provide connectors allowing for importing data from a variety of different databases, files, web services (both SOAP and REST-based) and APIs from selected web applications (mostly social networks or analytics software). Some solutions provide widgets that provide graphical interface to incorporate human activities into the workflow.

This overview shows that currently there is no solution that would provide developers with a lightweight cross-browser server-side middleware that provides web application inter-operability and considers user resources and roles management among distributed web applications. This situation motivates us to propose a platform that aims at introducing the said functionality. Its main goal is to simplify both integrated applications design experience and maximize their performance when in use.

2 Platform Architecture

The main influence for the Integration Platform architecture comes from the MVC software architecture pattern. This pattern, typical for many current web applications and application frameworks like .NET MVC 4.0, Ruby on Rails or Django, introduced a concept of separating the web application into 3 layers: Model, View and Controller. Each layer specializes in appropriate tasks: the Model is responsible for managing data for the application, the Controller for data processing and preparing them to be viewed and the View visualizes the data from the Controller building HTML structures dynamically. Other specialized standards have been gaining popularity with time, especially extending how the View is built: CSS style sheet language used for describing the presentation semantics of a document written in mark-up language and JavaScript language for dynamic View/DOM manipulation on the browser side.

Focusing on the MVC, we can observe that this layered approach did not necessary force the distribution of the application. In many aspects distribution of functionality has rather naturally come from integrating pieces of extra functionality provided by the web services – both in SOAP-based Web Service standard and even to greater extent using REST approach.

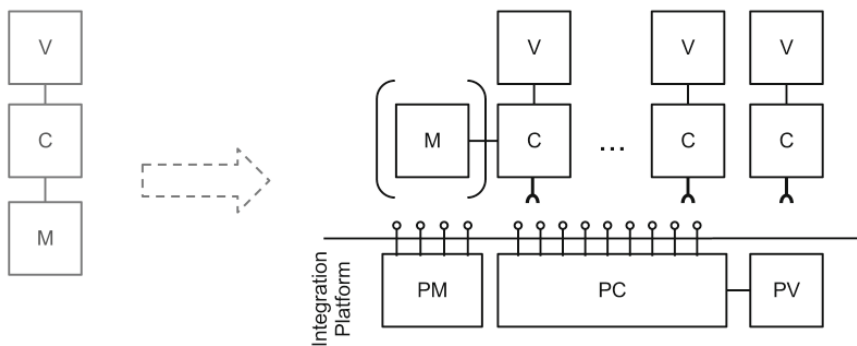


Fig. 1 Integration Platform extending the MVC software architecture pattern

The architecture proposed with the presented Integration Platform (Fig. 1) is based on two notions: that the MVC design pattern is vastly popular among web application programmers and, secondly, more and more web applications extend their functionality via web services. The former does not limit the application of Integration Platform to MVC-based web applications; however, its adoption rate would be significantly greater in those cases. The latter, when gathering more data on the subject, led us to an observation that adoption of web services in web applications is usually performed with no real design pattern in mind, leading in most cases to creating disorder in the framework of the MVC application itself. What is more, when utilizing web services, web application designers have no real control

over how they behave or how to monitor them. Granted, all those matters could be solved via self-written patch code but the same could be told about MVC-based applications and yet the strict pattern-based approach, enforced by the frameworks, allows for greater organization of code and functionality.

The Integration Platform manages web services and is prepared to organize them in thematic domains to be utilized by various web applications. Most of those services are in the Platform Controller, which is a well-organized abstraction (more on its details in Fig. 2), a collection of web service interfaces. Those could be any web services, also provided by other web applications but it is encouraged to implement them as stateless autonomous components, providing functionality via SOAP protocol. This way allows for more control and flexibility. The state for those services could be provided by the Platform Model, which is a collection of data providing services (standards use is encouraged, e.g. WS-Resource). Web applications could still use their Models for handling crucial data to be delivered quickly to their Controllers; however, if sharing data among distributed applications is the key requirement then web application Model could be omitted entirely.

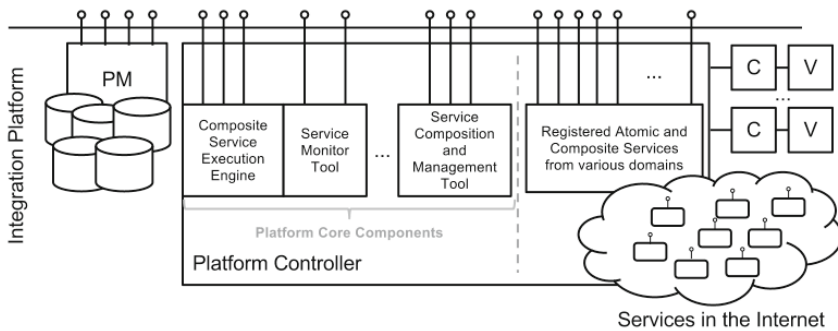


Fig. 2 Integration Platform overview

Figure 2 describes in more detail what are Platform Model, Platform Controller and Platform View abstractions. The Integration Platform delivers functionality mainly through web services that are distributed among preferably stateless Platform Controller Services and, to smaller extent, by the Platform Model data delivery services. The Platform Controller is in fact a set of services provided both by the Integration Platform users (mostly web applications authors) and the platform itself via specialized core components. Finally, the Platform View is a collection of web applications that provide Graphical User Interfaces for configuration and monitoring of the Integration Platform; those are not intended to be directly integrated with any of the external web applications.

3 Platform Core Components

As the Integration Platform main goal is to manage web services and provide functionality through web services many of its core components focus on those tasks and at the same time provide their functionality via web services.

Core components in general allow for composition of web services into complex processing workflows, registering those new composite services in the execution engine and managing their execution, relieving users of necessity to implement complex interaction among services and focusing on invoking a single service at a time. All those components operate on domain services provided by the user or services created inside the platform with service composition tools.

Platform View provides user interfaces for all core components, because most of them are server-side applications with only web service interfaces. The core components should be accessed directly via web services, thus increasing performance.

In the Platform a specific core component provided with a web based graphic user interface constitutes a distributed web application we call a tool. The basic tools created this way are:

- **Ontology management tool** – one of the key features of the platform involves following the semantic web principles to enable automated inter-application communication. The ontology management tool provides the support for ontology management, visualization and ontology grounding into concrete data types and resource types (described with XML Schema).
- **Service composition and management tool** – this tool provides a graphic interface for composite service definition as well as various automated service composition methods implemented as web services. The interface also enables users to define semantically annotated requirements for composite web services that can be composed by service composition engine embedded in the tool. Additionally, the tool enables developers to manage web services that are or can be used by web applications developed with the usage of the platform. This is quite a complex tool that integrates functionality of multiple components, e.g.: ontology manager, deployment services of the execution engine and even a set of composition services, which in this case could be treated as service-based toolbox.
- **Composite service execution engine** – this tool enables developers to execute their composite services created with service composition and management tool. This tool also allows developers to deploy their composite services to the execution engine. The deployment process creates a web service interface compliant with the WS-* standard which encapsulates composite web service and allows the developer to invoke it like a standard Web Service.
- **Service monitor tool** – this tool allows developers to keep track of all their services registered on the platform. The tool provides functionality that supports real time service monitoring, monitoring statistic gathering and statistic visualization. A unique feature of this tool is that it can monitor any service

externally, from the client perspective; this is performed via definition of specialized tests periodically and invoking the service. Not all services should be tested in this manner, especially tests on state-storing services could be performed only if they offer reverse operations and then they tests should be performed in collections with the use of transactions.

- Identity provider – this tool provides user identity to registered applications. It is based on OpenID and allows users to share one identity among various applications. With this solution a single sign-on feature can be offered to all applications developed with integration in mind. Implementing services from the identity provider in web applications enables the platform to support distributed resource access and roles management.
- Web application inter-operability manager – this tool provides a web interface, which allows for web application installation on the platform, offering roles and privileges management for both users and web applications accessing other applications services. It also provides a web API for web application interoperability management that enables resource management and resource retrieval by web applications, user privileges retrieval and direct download of java script components that can be embedded in applications to provide generic interface for switching between web applications deployed on the platform (system bar) and their interoperability (intent menu).

The last two tools show how the Integration Platform can offer services, which implementation broadens the initial concept of integration only via shared service-providing middleware external to all web applications, but allows to transparently share users, resources and services more horizontally – from other integrated web applications. This aspect of the platform builds upon its basic components data semantic description and integration, and provision of functionality via services. An example of such functionality use will be shown in the next section.

3.1 Platform Services

As it was stated before most platform tools that have been developed for the platform are of expandable nature and much of their functionality is provided by specialized replaceable services:

- ontology management tool uses specialized services for data format transformation and for reasoning purposes,
- for the service composition and management tool those are composition services that perform automated service composition with semantic-based service discovery and QoS-based service selection using optimization methods; the tool is equipped with set of several composition services, which can be easily expanded by registering new composition services,
- the composite service execution engine has a modular build, and the execution service component is in reality a specialized service for the interface component; this service can be replaced with different execution engine providing

different qualities; one of the engines that has been developed for the platform uses external web services to extend its functionality; those services provide ad-hoc interpretation capabilities e.g. dynamic alternative service selection when the current service is not responding,

- service monitor tool uses services to automatically built and deploy service tests in remote servers,
- each tool that needs to visualize a composite service uses a graphical user interface provided as a service.

4 A Basic Example of Platform Use in an Integration Scenario

Let us assume a scenario that hospitals are using a web application for patients' information management and this software have been integrated with a proposed platform. An ontology for the hospital domain has been defined in the platform and systems resources, this includes patients have been described with this ontology – namely patients are a type of *patient* resource. If one day a government approved company had introduced a web application or a web service that checks for patients' insurance validity then this service provided could be easily integrated with the hospital software. First, the application would have to register its capability i.e. a web service and a type of resource (here: a patient) that it operates on. Then, it would start working. The hospital application would not have to be further developed because the new functionality would be broadcasted to every web application that works on the referred resource type and each resource would gain a new operation from another web application.

The most important aspect is that those features are declaration based and can be managed by the software, ontology or integration architect and do not have to be implemented or designed into each application before a new service appears.

The Integration Platform also allows for applications not only to share services but also resources (two virtual hospitals can have a common patient base), accounts (patient can use the same identity, login etc. in many different web applications) or even create and use new composite services combined from services from various providers (and easily added to their web applications without implementation).

5 Note on Platform Implementation

The platform components have been developed in various technologies and integrated via web services:

- the Platform Model services deliver data from various databases – mostly relational and XML-based,
- the Service Composition and Management tool has been implemented using the Ruby on Rails framework and composition services were developed in Ruby and Java languages,

- current execution engine implementation uses Java and JAXB technology for interfaces management,
- the ontology manager tool, monitoring tool and many other platform components were developed using C# ASP MVC 4,
- the Integration Platform has been developed in C# ASP MVC 4 and uses mostly JavaScript for extending integrated applications capabilities; integrated applications implement the JavaScript-based graphical elements and communicate with the platform via AJAX and web services; external resources are delivered to web applications using REST approach.

6 Conclusions and Future Work

This work introduces a novel approach to web application integration, extending the MVC model of a web application to a platform integrating multiple such applications via web services. A brief overview of similar, complementary and competing approaches has been presented. In the next sections the platform architecture and its core components have been described in detail and their functionality briefly discussed in order to show how the platform supports web application development. Then, an implementation scenario was described to better understand how the platform was designed to be used.

To our best knowledge it is the first platform for delivering functionality via web service with such focus on web applications and OS-like mechanisms supporting them. The OS capabilities of the platform have been described with less detail as they are out of scope of the work; however, it is our goal to continue research also in this area. In future works more platform components will focus on better resource management and provisioning to web applications with anonymity and security guaranteed in those operations. It is planned that web applications will have no knowledge about other web applications and only about the resources with platform managing the authorization aspect. Also, in current development, the platform supports greater, yet still anonymous, integration among web applications via "horizontal" web services, which some applications provide to other applications.

Acknowledgments. The research presented in this work has been co-financed by the European Union as part of the European Social Fund.

References

- [1] <http://www.eyeos.com/> (accessed February 24, 2013)
- [2] <http://glidesociety.com/> (accessed February 24, 2013)
- [3] <http://osw3.com/> (accessed February 24, 2013)
- [4] <http://www.chromium.org/developers/web-intents-in-chrome> (accessed February 24, 2013)

- [5] Bangemann, T., Rebeuf, X., Reboul, D., Schulze, A., Szymanski, J., Thomesse, J.-P., Thron, M., Zerhouni, N.: Proteus: creating distributed maintenance systems through an integration platform. *Comput. Ind.* 57(6), 539–551 (2006)
- [6] Benslimane, D., Dustdar, S., Sheth, A.: Services mashups: The new generation of web applications. *IEEE Internet Computing* 12(5), 13–15 (2008)
- [7] Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American* (2001)
- [8] Brown, P.F., Hamilton, R.M.B.A.: Reference model for service oriented architecture 1.0 (2006)
- [9] Juszczyszyn, K., Świątek, P., Stelmach, P., Grzech, A.: A configurable service-based framework for composition, delivery and evaluation of composite web services in distributed qos-aware ict environment. *IBM Cloud Academy*, 123–134 (2012)
- [10] O'Reilly, T.: What is web 2.0: Design patterns and business models for the next generation of software. *Communications & Strategies* (1), 17 (2007)
- [11] Stelmach, P., Falas, L.: Service composer: a framework for elastic service composition. In: *Information Systems Architecture and Technology: Service Oriented Networked Systems*, pp. 47–60 (2011)
- [12] Świątek, P., Stelmach, P., Prusiewicz, A., Juszczyszyn, K.: Service composition in knowledge-based soa systems. *New Generation Computing* 30, 165–188 (2012)
- [13] <http://www.boomi.com/> (accessed February 24, 2013)
- [14] <http://www.snaplogic.com/> (accessed February 24, 2013)
- [15] <http://www.jitterbit.com/> (accessed February 24, 2013)
- [16] <http://www.mulesoft.com/cloudhub/ipaas-cloud-based-integration-demand> (accessed February 24, 2013)

Universal Platform for Composite Data Stream Processing Services Management

Paweł Stelmach, Patryk Schauer, Adam Kokot, and Maciej Demkiewicz

Institute of Computer Science, Wrocław University of Technology,
Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland
{pawel.stelmach, patryk.schauer, adam.kokot,
maciej.demkiewicz}@pwr.wroc.pl

Abstract. Constant delivery of data and information through streaming methods is growing in popularity. Streaming is widely used for video and sensor data delivery, usually directly to the client. In this work we propose architecture of the platform for composition of several distributed streaming intermediaries, able to process the data stream on-line. Features of the platform consist of ability to create a composite stream on demand as well as update, delete or read its current state. Works on the platform are an on-going research effort and current work, presented in this work, focuses on separation of platform functionality into distributed software components for performance optimization. This distribution allows for optimizing each component's behaviour regarding its usage characteristics. As an effect the platform's streaming service management functionality is offered as a stateless service.

1 Introduction

With the rapid evolution of the Internet, which has taken place in the last decade, many ideas appeared and changed the way we create web applications. Even the concept of web application itself has replaced a simple and static web-accessible HTML document. Readers have evolved into contributors and web pages became web applications but another greatly influential idea came with the Service Oriented Architecture paradigm and brought a concept of services, already widely known in business, to computer sciences. The notion of remote object access precedes the current century but the increase in Internet adoption and quality forced web services into public awareness. Then came standards, like XML-RPC, followed by WS-* for SOAP-based web services and WSDL for their description. Web applications ceased to be monoliths and more and more services started to extend their functionality.

The Web Service standard is based on request and result behaviour, often requesting some operation to be executed. After that the service is expected to stop working, waiting for the next request. This seems natural; however, some use cases require a different kind of behaviour. New user needs for continuous access

to ever-changing data, like video feeds or sensor data, require a different kind of service and put even more strain on the Internet transporting capabilities.

Streaming services can deliver audio/video surveillance, stock price tracing, sensor data ([7]) etc. With technology development both the size as well as the number of concurrent data streams has increased. On top of this we add middle-ware for on-line data stream processing ([1], [10]), changing video format or colour, calculating whether a patient had a heart attack (based on on-line heart rate monitoring) and more.

Distributed stream processing is becoming more popular, especially in domains like eHealth, rehabilitation and recreation fields where distributed measurement data acquisition and processing are indispensable ([11]). In case of such applications the processing time is crucial. Solution presented in [6] discusses the concept of distribution of processing services in a peer-to-peer computer networks in order to meet such requirements and, additionally, increase system availability and provide appropriate mechanisms for emergency handling. Also, data stream processing finds its use in computational science or meteorological applications, where there are multiple data sources and multiple recipients interested in the processed data stream ([8]).

From the architectural standpoint multiplication and distribution of data stream processing services in fact does not change much the already complex problem of data stream routing. Both data sources as well as endpoints could potentially be distributed and, assuming that processing services could be more atomic and composable ([12]), such approach brings more flexibility. Considering the geographical distribution of data sources and endpoints, distribution of data processing could be an optimal solution in many cases, especially if it had minimized the size of data being transferred over large distances at certain steps of the composed service. Individual data streams could be routed to subsequent processing services, where appropriate calculations would take place without braking the flow of the stream ([2], [9]). This approach allows for increasing performance, especially when streams should be processed in real-time ([1]) and each atomic service could be optimized for a specific processing task.

For this purpose we propose a platform called the ComSS Platform (abbreviation for COMposition of Streaming Services) that addresses the requirements mentioned above and offers management functionalities that allow for creating, reading the state, updating and stopping compositions of any data stream processing services, provided that they are compatible (that is each two services communicating have to be able to communicate via a common protocol).

2 Platform Description

2.1 Platform Goals

The main goal of the platform is to manage data stream processing composite services (also called composite streaming services). Composite service is a collection

of several stream-processing services that typically operate on a single data stream. Each service transforms data (ranging from simple data manipulation – like determining average – to complex anomaly detection algorithms and more) and transfers it to the next service in a composition. For example a colour video stream can be transferred to a service that transforms it to a black and white video and then to a service that detects objects in a video (which could be more efficient with black and white video).

This simple example is based on two services in a series structure but composite services can be more complex: they can process multiple data streams and be described by complex workflows and not only series structures. Also, although composite service can merge two streams into a single data stream, this is not their sole purpose and composite streaming service (or composite data stream processing service) should not be confused with stream composition.

Provided the designer of streaming services would like to utilize the platform automated management capabilities, he can delegate all the tasks of assembling, disassembling or monitoring of such services to the platform, simply by using its basic web interface.

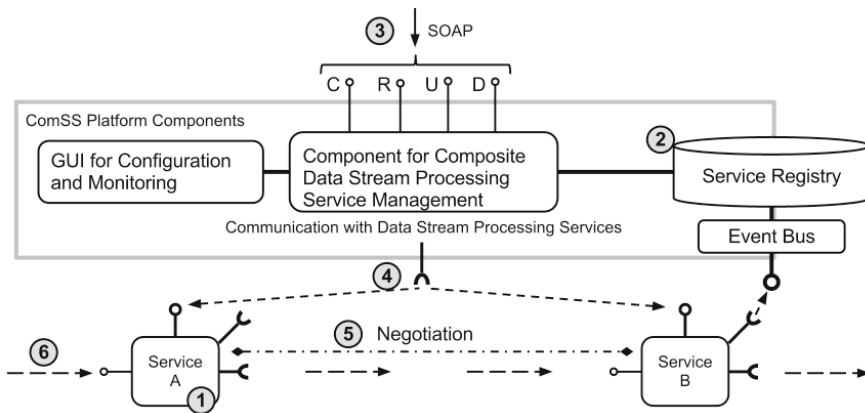


Fig. 1 Overview of the Data Stream Processing Service Framework

In more detail, the ComSS Platform capabilities are mostly based on CRUD-based web interface, allowing for a specific composite service (treated as a resource) management; those are:

- creating a new composite streaming service based on a user request, containing a set of atomic services to be integrated,
- reading data about the current state of the composite streaming service execution,
- updating the composite streaming service, given new parameters for communication or replacing some of its atomic services,
- stopping and deleting the composite streaming service (not the atomic services themselves but their configuration and instances created for this composition).

Additionally, the platform is provided with internal components responsible for events distribution among its components and services, resources registration (especially atomic and composite streaming services) and communication with and among atomic streaming services during composite service control sequences (creation, reading, update, deleting).

2.2 Platform Overview

The ComSS Platform consists of several autonomous software components but also requires atomic streaming services (external to the platform, provided by the client) to implement specific communication libraries for control purposes. To facilitate implementation, those libraries are provided with a programming framework, which will be described in the next section.

In Fig. 1 separate boxes represent platform components: GUI, composite service management (e.g. service composition), communication with data stream services (negotiation supervisor), Event Bus and Service Registry and – outside of the platform – data stream processing services. We wanted to highlight particular service interfaces (line with a circle) and service calling capabilities (lines with unfinished circle) in some of those components when they are crucial to the life-cycle of the composite streaming service. Straight lines show that some components are interconnected but details about what are those internal connections and interfaces are out of scope of this work.

Components that are part of the platform are autonomous and communicate via before mentioned web services with SOAP protocol. Notice that the SOAP protocol is used only for control aspect of the platform, e.g., for communication protocol negotiation or starting and stopping of a composite streaming service. The streaming service itself is transferring data continuously through a data stream and uses a proprietary protocol that has been negotiated beforehand. In Fig. 1 only two streaming services are shown as an example of a simple composite service and the data stream is considered to come from an external source – not visible for the platform itself.

The platform itself delivers to its clients a simple web interface to create, read, update and delete a composite streaming service. Rest of the interfaces were designed for internal use; however, Service Registry and Event Bus could be provided from outside of the platform, allowing for sharing of resources and thus having their own external web interfaces.

The main assumption for the ComSS Platform was to focus on performing operations, which were fully defined using the CRUD-based web interface, exactly when they were needed and shutting down when they were completed. Such stateless behaviour has been introduced to minimize the chance of error during runtime and to shift the responsibility for service performance to the streaming services themselves. Following this concept was the decision to register composite services only for identification and logging purposes, so that the ComSS Platform could support the composite service only when requested, for instance the platform

reacts to errors reported by the streaming service. For that purpose the Event Bus, (the always-up component) is listening for events coming from atomic streaming services informing about errors or exceptions needed be handled, both their own or based on the unexpected behaviour of other services in the composition.

The basic scenario for the ComSS Platform is to create a new composite streaming service given a graph of atomic streaming services. It is assumed that those services have been implemented using the provided framework or implement necessary libraries (fig 1.1) and are registered in the service registry (Fig. 1.2). This step is not necessary but is useful for logging and identification purposes or when dynamically searching for service candidates if user request was not precise enough or requested services are not responding. For user *create* request (fig 1.3) the ComSS Platform searches for appropriate atomic streaming services and requests their participation in the composite service (Fig. 1.4). Using SOAP protocol the Platform informs each of the atomic streaming services about their immediate neighbours in the composition (e.g. next in a sequence), with which they will have to negotiate the communication protocol.

Streaming services start negotiating (Fig. 1.5) and create new service instances to handle the new composite service request. Each physical service is actually a server that can handle multiple requests and multiple streams – one atomic service can be a part of various composite services. To avoid confusion in the implementation, parts of the service that handle different streams are separated from each other by creating instances. Physically it is still one service, but for the purpose of identification in the system, each service instance can be treated as a separate service with different id, address and port etc. but with the same functionality and quality. Using the concept of service instances we can assure that each service (instance) is used only in one composite service and data streams are always correctly separated and transferred among services in a single composite service.

Finally, the ComSS Platform finishes its work and the streaming to the composite streaming service can start.

3 Streaming Service Framework

Part of the effort to make the streaming service composable lies with the service designer himself. He has to follow conventions for the streaming service design and implement necessary libraries for control, negotiation and communication. Using the framework provided with the ComSS Platform allows him to focus on implementation of the data stream processing algorithms alone (Fig. 2).

It should be noted that – in contrast to Web Services using the SOAP protocol – streaming services are not limited to a single protocol. The proposed framework allows for the use of any protocols for communication; however, it uses web services for control over the atomic streaming services. With web services the ComSS Platform conveys requests for creating new instances, reading their status or updating their parameters and, finally, deleting the appropriate instance.

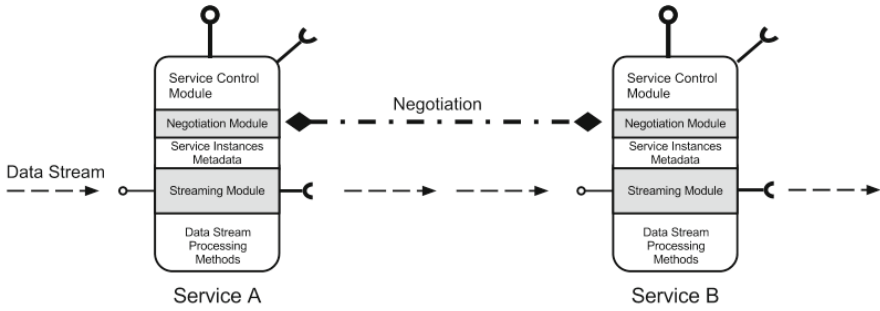


Fig. 2 Overview of the Data Stream Processing Service Framework

The atomic streaming services also can send SOAP messages, mostly error or exception messages for the Event Bus to handle, but the designer can also implement other behaviours for scenario specific purpose – like email sending request etc. Some capabilities could be implemented in each streaming service, but they could be outsourced and shared via a web service, preferably using the Event Bus to execute specialized web services when necessary (which could implement various scenarios, like stopping or recomposing the composite streaming service).

In general, streaming services perform two types of communication.

SOAP-based communication, which is used for:

- sending a negotiation request (streaming service is delivered a list of neighbour services in a composite service with which it will start a communication protocol negotiation process),
- creating a new service instance (this is not to be confused with starting the data stream, because after creation each service is *ready* to receive, process and transfer the data stream and actually awaits for the stream to be transferred from external sources),
- control over the service when it is running (transferring and processing data stream) – reading its state, stopping it (deleting a service instance),
- standard error messages (events) propagation when services express unwanted behaviour,
- personalized messages (events) propagation, when service designer deliberately implemented such behaviour (sometimes sending a message via a Web Service is more natural then sending it via stream – e.g. when an anomaly is discovered then alert is a single event that should be directed to a specific recipient that can react to it, and not transferred with the stream).

Transferring the data stream:

- there are many different data stream formats and communication protocols and each service can support a selection of those solutions,
- it is the negotiation phase task to determine which protocol, format, port etc., will be used for the data stream transfer,

- the data stream is transferred independently of the control messages transfer (via Web Services) – usually the exchange of control messages takes place before the data is streamed or can end the stream transfer (requesting its deleting); during the data stream the Platform should be offline and only respond to errors or special events handled via the Event Bus.

In the basic scenario of creating a new composite streaming service, a request for a new service instance is sent to the streaming service via the web service interface (Fig. 2) in the negotiation phase. The framework communicates with neighbour streaming services, indicated by the ComSS Platform as next or previous in the composition. If the service confirms that it knows the requested protocol, gives an address and opens a port for communication, then a new atomic streaming service instance can be generated.

Then, when all services are ready to communicate, first messages are sent to the streaming interface of the streaming service. The role of the streaming module is to receive the data stream and prepare it for processing, which is de facto the function-providing part of the service. Next, data is prepared to be sent to the appropriate address and port of the next atomic streaming service in the composition.

4 Relation to Previous Works

Work presented in this work is part of an on-going research on a Universal Communication Platform ([3], [4], [5]). As a result of that research streaming services composition mechanisms were proposed in a prototype form. Our goal is to out-source those tasks to a specialized platform and extend its scope to end-to-end composite streaming service management. Compared to the earlier, prototype version a greater focus was put on performance of the service management layer as well as the performance and method of creating instances of data stream processing services. A complete separation of the service composition and usage – making them time and physically independent (by loose coupling of interfaces) – increased the flexibility and scalability of the system. The new version of the platform is stateless and mostly active only on demand, which results in better resource management. However, in effect the composite streaming service is not directly monitored during execution. With the ComSS Platform libraries implemented in streaming services, each service can report on its own state as well as on the state of neighbour services, especially if they behave contrary to the expectations, but the platform does not actively request those reports and if some services stop working it is up to their neighbours to report this behaviour.

Another major improvement lies in using external services in various management scenarios, mainly during emergency and unpredicted situations. Web services are used for recomposition or replanning purposes but it is planned to further develop this mechanism to use automatically composed reactions to emergency scenarios, based on user preferences, system policies and current situation.

5 Example of Platform Use

Consider a simplified example that some developer has built a streaming service that detects specific objects or people in a video stream. He has noticed that the algorithm performs better when it is provided with a black and white video. He could further develop his service so that it would change the video signal to black and white but two things stop him from doing that: one the service would be too specialized that he would want it to be, because perhaps someday he would want to analyse video in colour and, more importantly, he has found another service that already does what he needs. What he has to do is to somehow connect those services. But he cannot access the other services' code to statically implement his service address or communication protocol and in fact he should not do this.

Using the ComSS Platform he only has to point which services should be combined in a composite service and used in his specific scenario. The protocol negotiation, starting and stopping of those services will be handled by the platform with no need for additional code implementation. Also, with no direct addressing (and creating instances) those services can services multiple purposes and after this purpose is fulfilled no legacy code (statically connecting one service to another) remains.

The above example is simplified and intentionally omits matters concerning data transport through the network (which in case of video will introduce delay). In fact, with this platform and our on-going work on service composition and optimization such services could be discovered and composed dynamically and after automated consideration of various scenarios non-functional properties (service and transport costs, delay, etc.) an optimal solution can be selected and provided as a single service (treated as a black box).

6 Conclusions

This work presents a platform for distributed data stream processing services management. All its software components have been discussed from an architectural perspective, describing how the platform realizes its basic goals, which are lightweight and stateless composite services creation, monitoring, update and deletion. The framework for atomic streaming services enables service communication protocol negotiation and error reporting to the platform via the Event Bus.

Acknowledgments. The research presented in this work has been co-financed by the European Union as part of the European Social Fund and within the European Regional Development Fund programs no. POIG.01.01.02-00-045/09 \& POIG.01.03.01-00-008/08.

References

- [1] Chen, L., Reddy, K., Agrawal, G.: Gates: a grid-based middleware for processing distributed data streams. In: Proceedings of the 13th IEEE International Symposium on High performance Distributed Computing, pp. 192–201 (2004)
- [2] Frossard, P., Verscheure, O., Venkatramani, C.: Signal processing challenges in distributed stream processing systems. In: Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2006, vol. 5, p. V (2006)
- [3] Grzech, A., Juszczyszyn, K., Świątek, P., Mazurek, C., Sochan, A.: Applications of the future internet engineering project. In: 2012 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel Distributed Computing (SNPD), pp. 635–642 (2012)
- [4] Grzech, A., Rygielski, P., Świątek, P.: Translations of service level agreement in systems based on service-oriented architectures. *Cybernetics and Systems* 41(8), 610–627 (2010)
- [5] Grzech, A., Świątek, P., Rygielski, P.: Dynamic resources allocation for delivery of personalized services. In: Cellary, W., Estevez, E. (eds.) *Software Services for e-World. IFIP AICT*, vol. 341, pp. 17–28. Springer, Heidelberg (2010)
- [6] Gu, X., Nahrstedt, K.: On composing stream applications in peer-to-peer environments. *IEEE Trans. Parallel Distrib. Syst.* 17(8), 824–837 (2006)
- [7] Gu, X., Yu, P., Nahrstedt, K.: Optimal component composition for scalable stream processing. In: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, ICDCS 2005, pp. 773–782 (2005)
- [8] Liu, Y., Vijayakumar, N., Plale, B.: Stream processing in data-driven computational science. In: 7th IEEE/ACM International Conference on Grid Computing, pp. 160–167 (2006)
- [9] Rueda, C., Gertz, M., Ludascher, B., Hamann, B.: An extensible infrastructure for processing distributed geospatial data streams. In: 18th International Conference on Scientific and Statistical Database Management, pp. 285–290 (2006)
- [10] Schmidt, S., Legler, T., Schaller, D., Lehner, W.: Real-time scheduling for data stream management systems. In: Proceedings of the 17th Euromicro Conference on Real-Time Systems (ECRTS 2005), pp. 167–176 (2005)
- [11] Świątek, P., Klukowski, P., Brzostowski, K., Drapała, J.: Application of wearable smart system to support physical activity. In: *Advances in Knowledge-based and Intelligent Information and Engineering Systems*, pp. 1418–1427. IOS Press (2012)
- [12] Świątek, P., Stelmach, P., Prusiewicz, A., Juszczyszyn, K.: Service composition in knowledge-based soa systems. *New Generation Computing* 30, 165–188 (2012)

Proposal of Cost-Effective Tenant-Based Resource Allocation Model for a SaaS System

Wojciech Stolarz and Marek Woda

Instytut Informatyki, Automatyki i Robotyki, Politechnika Wrocławska
voytec0dh@gmail.com, marek.woda@pwr.wroc.pl

Abstract. Software-as-a-Service (SaaS) is a software distribution paradigm in cloud computing and represents the highest, software layer in the cloud stack. Since most cloud services providers charge for the resource use it is important to create resource efficient applications. One of the ways to achieve that is multi-tenant architecture of SaaS applications. It allows the application for efficient self-managing of the resources. In this paper the influence of tenant-based resource allocation model on cost-effectiveness of SaaS systems is investigated. The tenant-based resource allocation model is one of the methods to tackle under-optimal resource utilization. When compared to traditional resource scaling it can reduce the costs of running SaaS systems in cloud environments. The more tenant-oriented the SaaS systems are the more benefits that model can provide.

1 Introduction

One of recent solutions for over- and underutilization problems may be a tenant-based resource allocation model (TBRAM) for SaaS applications. That solution was introduced and tested with regard to CPU and memory utilization by authors of [8]. They proved the validity of TBRAM by reduction of used server-hours as well as improving the resources utilization. However, the authors deployed their solution into a private cloud, which can only imitate public cloud environment. They tested cases with incremental and peak workload of the system. In this paper it was waned to check whether the TBRAM is really worth to adhere.

Examining that system in public and commercial cloud environment could deliver the answer for that question. Therefore, the main aim of the paper is further TBRAM approach validation, as it was proposed in future research part of the base work [8]. If the results of the research will confirm usefulness of the model then it could be considered as the solution for previous mentioned provisioning problems.

Despite that in the cloud one can automatically receive on-demand resources one can still encounter problems related to inappropriate resource pool at the time. These are over- an underutilization which exists because of not fully elastic pay-per-use model used nowadays [17]. Over provisioning exhibits when, after receiving additional resources (in reply for peak loads), they are being kept even if they

are no longer needed. Thus we are affected from underutilization. Under provisioning (saturation) exhibits when we cannot deliver required level of service because of insufficient performance. This is also known as an overutilization. It leads to the customers' turnover and revenue losses [3]. For example Amazon Elastic Cloud Computing (EC2) service charge users for every partial hour they reserve each EC2 node. Paying for server-hours is common among cloud providers. That is why it is very important to utilize fully given resources in order to really pay just for what we use.

We are still in early stages of cloud computing development. No one shall expect cost effective pay-per-use model for SaaS application after just deploying it in the cloud. What is more, automatic cloud scalability will not work efficiently that way [20]. To achieve desired scalability one need to design a SaaS application with that in mind. In order to do that, the application must be aware how it is used [11]. One can use multitenant architecture to manage the application behavior. It allows using a single instance of the program by many users. It works in similar way like a singleton class in object programming languages, which can supervise creation and life cycle of objects derived from that class. Supporting multiple users is very important design step for SaaS applications [5]. One can distinguish two kinds of multi-tenancy patterns: multiple instances (every tenant has got its own instance running on shared resources) and native multi-tenancy (single instance running on distributed resources) [2, 5]. First pattern scales quite well for small number of users, but if there are more than hundreds it is highly advisable to use the second one.

1.1 Overutilization

The term "point of exhaustion" is often used in relation to overutilization. It is described as a point when some resource is fully utilized, for example 100 % CPU usage or all memory is consumed [16]. This definition tends to be accurate in many simple cases. However, in case of cloud computing that definition seem to be an oversimplification. Authors of [12] propose another definition, according to which the point of exhaustion is a maximal payload that can be assigned to a single virtual machine without decreasing its throughput in the same time. Readings above the exhaustion point describe a saturated machine. This new exhaustion point definition requires measuring a VM throughput together with the resource utilization (CPU or memory). Authorial test SaaS system uses the JMeter tool combined with CloudWatch network related metrics in order to calculate that. The system is stressed with HTTP requests generated by the tool. Then it calculates the throughput of the system by dividing the number of HTTP request by the time from start of the first request to the end of the last one. By measuring it this way we can include all the processing time between the requests as well. In previous works [12, 13, 15, 16, 18, 19] authors focused on throughput to discover inflection points. Whenever the throughput was dropping while the VM utilization was rising an inflection point was found. In this work the same approach was used.

Authorial SaaS system used above mentioned inflection points to detect an overutilization of given virtual machine. All the VMs with Tomcat are monitored by gathering resource utilization metrics and throughput. The metrics used are as follows: CPU usage and Java virtual machine heap memory consumption. Based on that measures it can be told with good accuracy whether a VM is saturated in given moment or not.

Generally, when the VM is saturated the operating system processes start to use more and more resources making user's processes execution even slower. It has a negative influence on system responsiveness and therefore, on user experience. That is why it is always a good idea to avoid saturation. Even despite it does not have a direct influence on cost (we do not pay extra for high VM usage rate), it can drive to users turnover because of a poor performance.

1.2 Underutilization

From the economical point of view underutilization is just a waste of money. It means that we pay for something we do not need or not even use. From the end user perspective it is hardly noticeable, so from the provider perspective this extra money is spent almost on nothing. More formally, from the definition [4, 7] an underutilization describes a situation when some of the cloud resources are not being used by the working virtual machine. Off course it is almost impossible to assure 100% resource usage all the time so some kind of underutilization is inevitable. Underutilization in a cloud can be measured by the amount of the resources available for use. According to [4] resource is wasted when we can reallocate given resource utilization into another VM without exceeding its maximal quantity allowed. It means that for example: the payload for two VMs could be easily allocated just in one VM making the other VM unused. In order to check if given VM can be allocated to another one we need to calculate combinations of VMs according to some resource. One way to solve that problem is by using the knapsack algorithm. As proposed in the base article [8], the amount of used resources to the knapsack items' weights was assigned. As a value of an item the available quantity of the resource in other VMs was taken. In case of Java heap memory the item's value equals an amount of heap memory that still can be used (available memory). Thus, the most valuable items are the less used ones. The capacity of the knapsack is the amount of available resource of a VM we try to assign the workload to. By using this approach we obtain the maximum number of VM than can be potentially released. That VM number is used to measure underutilization. The lower that number is the better the resources are used.

1.3 Cost

Running the system in a public cloud gives us yet another way to assess the cost effectiveness. Almost every action made in cloud is registered and added to our bill. We pay for sent Internet requests, storage, VM hours and many more.

Therefore, the billing statement yields arguably the most accurate estimation of cost-effectiveness. At the end of the day it is the price we need to pay for our cloud service. During the tests the Amazon CloudWatch service was collecting metrics about the cloud environment usage. Both SaaS systems (Base System and TBRAM) are tested against the same test plan, so the requests number is exactly the same. The main difference between them can occur in the number of used virtual machines. That difference should be reflected on the bill statement. The comparison of costs of running the SaaS systems will show if there is any economic improvement with using the TBRAM approach over the traditional resource scaling approach.

2 Related Work

Authors in [5] propose profiles approach to scaling in the cloud. They try to use best practices and their knowledge in order to create scalable profiles. The profile contains information that helps to characterize a server in terms of its capabilities. When the scaling activity is fired it takes the profile information into account. In [9] authors propose a toolkit using Java mechanism to support multi-tenancy. They use context elements to track applications running on Java Virtual Machine. That in turn allows distinguishing each tenant. That information can be later used in order to estimate given tenant's resource usage. The tenant context can also be used for billing each tenant's activities. In [6] authors consider an intelligent resource mapping as well as an efficient virtual machines (VM) management. It is a very important problem that greatly influences costs of running applications in a cloud. In [10] authors describe three major components which influence virtual machines performance. These are: measurement, modeling and resource management. They introduce a decomposition model for estimating potential performance loss while consolidating VMs. Amazon proposes its Auto Scaling tool [1] to manage VM instances using predefined or user-defined triggers. It is the Amazon EC2 platform specific mechanism based on resource utilization. In [8] authors implement a tenant-based resource allocation model for their SaaS application deployed in private Eucalyptus cloud. The authors performed tests with incremental and peak workload simulation. In the research they achieved significant reduce of server-hours compared to traditional resource scaling model. The tenant-based model improved also utilization of cloud resources by their SaaS system. Moreover, they introduce formal measures for under and over provisioning of virtual resources. The measures are designed specifically for SaaS applications with respect to CPU and memory utilization. In this paper cost-effective tenant-based resource allocation model of SaaS system is presented. It will be referred as the base based system.

3 System Design

The TBRAM consists of three approaches that leverage multi-tenancy to achieve its goals. The first of them is tenant-based isolation, which separates contexts for different tenants. It was implemented with tenant-based authentication and data persistence as a part of the SaaS platform (Tomcat instances). The second way is to use tenant-based VM allocation. With that approach it was able to calculate the actual number of needed VMs by each tenant in given moment. The last but not least is the tenant-based load balancing (due to paper length it won't be described). It allows distributing virtual machines' load with respect to certain tenant. An overview of the architecture is presented in Figure 1. The dashed line in the picture denotes communication to web services. We can notice that the SCWA element in the Figure was the only change made to the original test bed [10]. That element embraced proposed TBRAM approach.

3.1 *Tenant-Based Isolation*

To assure that system worked properly it needed to isolate one tenant from another. A situation when one tenant can access and affect data that do not belong to him/her is unacceptable in any commercial solution. The TBRAM approach proposes low level isolation as it improves its scalability [2]. The tenant-based isolation of TBRAM could be split into two implementations. One was based on data persistence and the other one was based on authentication mechanisms. In this place is worth to mention that tenant based isolation was also used in the Base System. That was because both systems were using the same multi-tenant database. What is more that technique was practically affecting only the SaaS platform, so it is isolated from the SCWA concept. Thanks to that in both systems the SaaS platform was exactly the same, thus minimizing its influence on the results.

In the persistence layer the authors propose Shared Database – Shared Schema as it has the lowest hardware cost and the largest number of tenants per server [14]. To logically separate data the Tenant ID field is used for each database table. From technical point of view JoSQL libraries were used, which let to perform SQL-like queries over Java collections. Those libraries were used by Struts2 interceptors to achieve multitenant preprocessing. Java annotations were used to mark the places in code that needed this kind of tenant-based behavior. That was arguably the most efficient way to implement multi-tenancy since the data were first fetched and then filtered. It could be achieved using SQL selection mechanisms. Interceptors as an implementation of aspect oriented programming postulates had many advantages as well. The main was that all the code was in one place but could affect any class marked with the annotation. Secondly, that annotation was the only change that needed to be made to an existing application code to enable multi-tenancy. Therefore it could possibly be the most common way to add that tenant layer to existing applications.

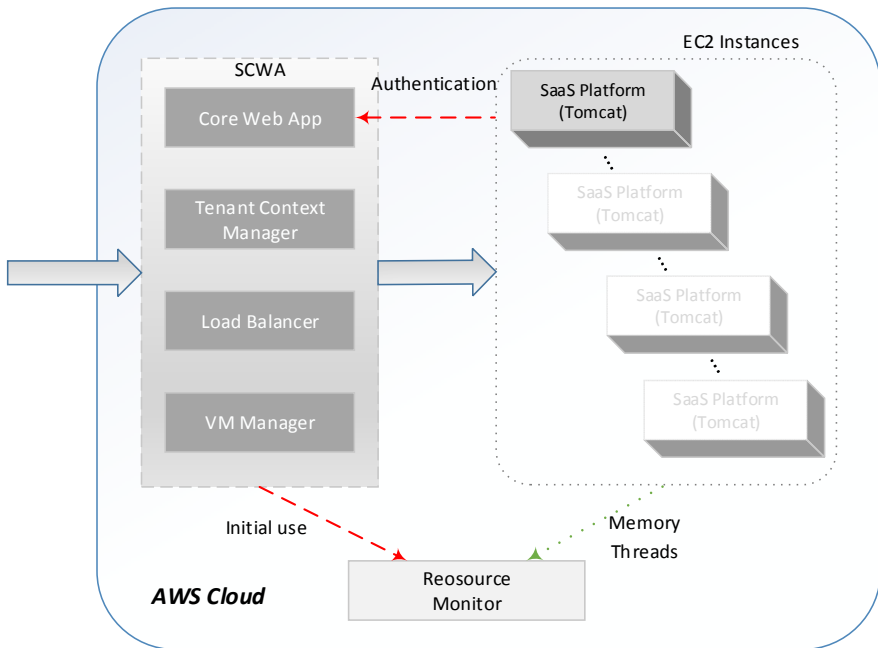


Fig. 1 TBRAM system architecture

Tenant-based authentication was the second concept used to achieve tenants' isolation. As proposed by TBRAM it should be implemented into the core application of the system which is SCWA. During the authentication every user was linked to its Tenant ID. From now on the user could access only the data the certain tenant has rights to. Needless to say that one could not access any data before the authentication. The TBRAM also suggest using an Access Control Lists (ACL), which it was decided to omit as it introduces just unneeded complication for me. It was decided to give full access to all SaaS applications to all users for simplicity. It was necessary to receive the tenant information from any point in the SaaS system. The TBRAM proposes a mechanism based on cookies and the servlet context. The authors [8] used a local Tomcat cluster to deploy their solution. In my case the SaaS system was deployed into the Amazon cloud infrastructure and that solution did not worked for me. It was decided not to use Tomcat instances running in cluster mode. That was because an overhead related to sharing session information between all cluster's nodes was a concern. If the nodes are running in different networks it was thought that it can introduce non negligible influence.

Currently, Tomcat 7 version supports all-to-all session replication. It was used a special web service to serve the tenant information instead. It was more platform-independent and it could work in both cases. Whenever any user was accessing given VM for the first time, the SaaS platform was checking in SCWA if that user

was authenticated and authorized to do that. If it was, then its specific data were saved locally in a session context so the next request from that user didn't require further communication to the SCWA's centralized web service. Therefore, only VMs that needed that specific information were acquiring it. There are also other methods of session replication like session persistence (shared file system or database) which are outside the research scope.

According to TBRAM a Tenant Context object was conceptualized. It contained information about tenant ID, active users and their VM assignments etc. A Tenant Context Manager object in turn was used to manage all the underlying Tenant Context objects. Thanks to that information about the tenant's state was available to all other services. The Tenant Context allowed isolating each request sent to the platform based on given user's tenant information. We can see several users from two different tenants (subscribers). Despite they physically share the same SaaS applications and the persistence layer they are still logically isolated by their tenant contexts. These context objects help to achieve native multi-tenancy of the applications. The users have no idea they are sharing the same resources.

3.2 *Tenant-Based VM Allocation*

Tenant-based VM allocation was used to determine the number of VM instances needed for given tenant in given moment. It combined the concept of profile approach with monitoring services implemented within the SaaS system. A profile used for the test bed in the base paper was a small virtual machine profile. It was meant to substitute the m1.small EC2 instance in Amazon cloud. The profile was as follows: 1 CPU core, 1 GB of RAM, 800 MB to the JVM heap memory and 100 as the number of users the VM can handle ($Threads_{max} = 200$). In this work similar profile was used since the SaaS platform was deployed in actual m1.small instance. The main difference was the maximal number of users set to 50 in that case. This profile information together with current readings from metering services was used to calculate required number of VM instances.

The Tenant Context Manager was responsible for assigning the weights to each Tenant Context. These weights were later used for VM calculations. The TBRAM proposes the following formula:

$$TenantContextWeight = users_{active} * (heapSize/Threads_{max}) \quad (1)$$

where active users are those whose session has not expired. Heap size is the amount of memory assigned to JVM (set in profile) and the $Threads_{max}$ is a maximal allowed number of concurrent threads for the SaaS platform. The fragment in parenthesis could be treated as an average memory usage per thread for given profile. Therefore the formula above is an estimation of required memory for given number of active users. The second formula is used to calculate the VM capacity:

$$capacity_{VM} = heapSize - ((heapSize / Threads_{max}) * Threads_{platform}) \quad (2)$$

The formula subtracts current memory usage from the maximum allowed amount described in the profile. The current memory consumption is calculated by multiplying number of SaaS platform's threads (when in idle) by the average memory per thread.

From this formula we know how much memory is available solely for users of given SaaS platform. This is because from the amount of memory assigned to JVM some part is consumed by the Tomcat's and SaaS platform's threads just to start the service. All the following threads were created to serve each user. Thanks to that it was able to estimate actual initial resources available.

The TBram suggest use of a knapsack algorithm to calculate the minimum number of instances needed to allocate current workload. This number was the only result yield by the algorithm since it was not interested in actual tenants' assignments to available VMs. The algorithm used the values returned by the above formulas (Formula 1, 2). Dynamic programming method was used to solve the knapsack problem quickly. This whole idea was conceptualized within Tenant-Based VM Calculator. The results of these calculations determined the number of VM instances requested from AWS cloud by the VM Manager. So the first source of information about needed number of instances came from knapsack algorithm. Yet, it was not the only one. Sometimes even the most advanced estimations are inaccurate, thus leading to discrepancy between reality and its state kept by an application. That is why it was decided to add also user factor. If several subsequent request dispatches failed then a new VM instance was requested from VM Manager on user's behalf.

4 Preliminary Test Results

This chapter presents the results of conducted initial tests. It presents the results in terms of server-hours, over- and underutilization as well as the cost.

When the tests were over it was time to collect measured data. All of them were gathered by Amazon CloudWatch monitoring service. That tool allows viewing some basic statistics of data in form of charts. However, in order to perform more advance analysis it was needed to download the raw data for further processing.

On the chart (Figure 2) we can see the comparison of the Base System and the TBram system in terms of combined resource underutilization. The results come from the incremental workload simulation tests. We can notice that during the first four months of simulated year the utilization problem did not exist in case of the TBram system. In the middle of the year the systems are comparable, but at the end of the year the Base System was significantly more efficient. In overall we can say that in case of the incremental tests both systems yield approximately the same resource waste (underutilization).

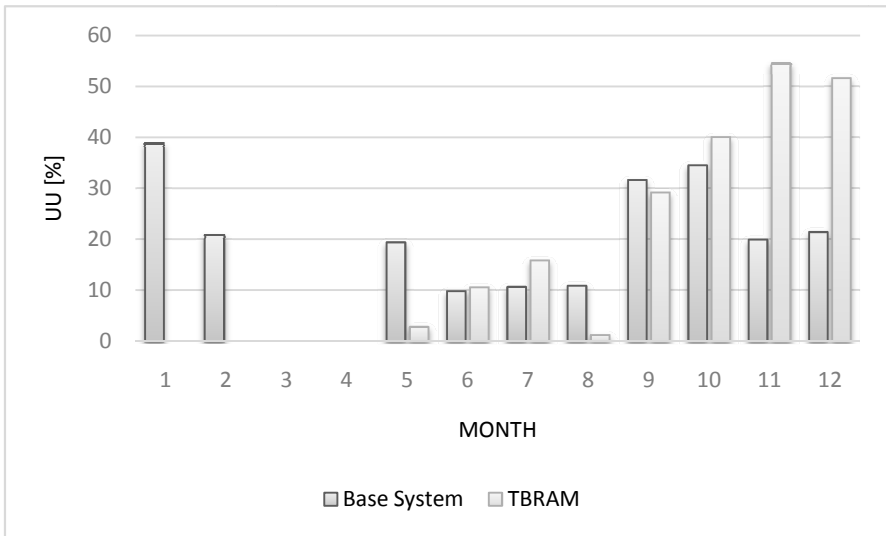


Fig. 2 Resource underutilization during the incremental workload tests of the systems

Now let us compare the systems in case of the peak-based workload tests. The following chart presents the results (Figure 3). Now the improvement from using the TBRAM is clearly visible. For most of the simulated year the TBRAM system was much more efficient than the Base System in terms of combined resource underutilization. Often the improvement was by 50% and more. The dynamic VM fleet tenant-based management showed its superiority when the workload was rapidly changing.

It is important to notice that both averages for %UU are generally lower than in case of traditional scaling system. But, in order to confirm that one average is statistically different than the other a t-student test was used. In our case we wanted to check if the TBRAM system was a significant improvement to the Base System.

Apart from the CloudWatch data one could also assess the systems on a financial basis. The Amazon's billing **statement** together with the AWS Simple Monthly Calculator was the data sources for the economic cost analysis. It was interested in the difference between the systems cost more than in total cost itself. Therefore in cost comparison it was taken into account only the parts that differs the both systems.

That was the size of SaaS platform VMs fleet and the type of load balancer. All other test bed parts like the database, RCM or JMeter cluster were excluded from the price comparison. Since the both test beds were stressed with exactly the same workload by performing the same test plan, the usage of database and other resources was the same. It was thought that excluding these elements can only increase the clarity of such a comparison. Finally, in the comparison it was taken into account only the cost of EC2 instances with the SaaS platform and the load balancers. Below Amazon EC2 prices for EU (Ireland) region for 2012 are presented.

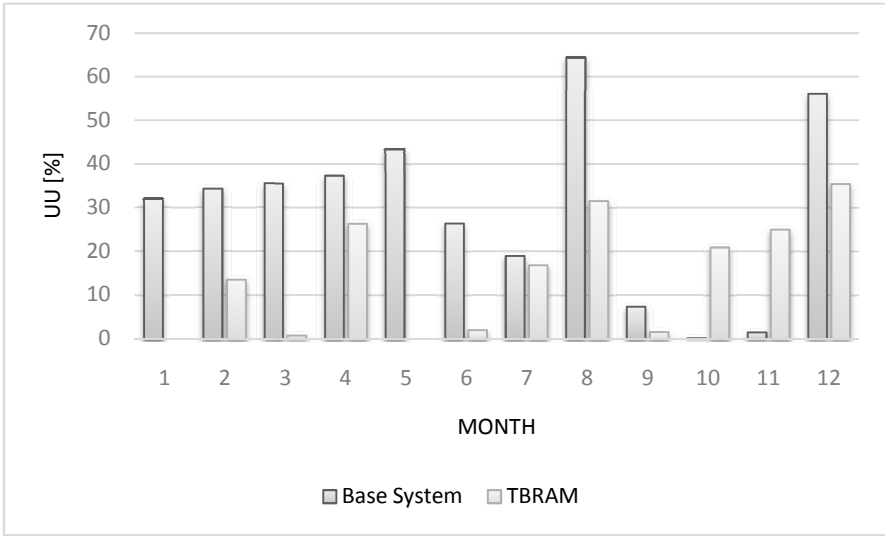


Fig. 3 Resource underutilization during the peak-based workload tests of the systems

Table 1 AWS EC2 cost of on-demand instances in EU

	EC2 INSTANCE TYPE		
	<i>m1.small</i>	<i>m1.medium</i>	<i>ELB</i>
Cost per hour	0.085 USD	0.170 USD	0.028 USD

Table 2 AWS EC2 data transfer cost in EU region

	DATA TRANSFER TYPE		
	Transfer out	ELB in/out	ELB data proc
Cost per GB	0.120 USD	0.010 USD	0.008 USD

Table 1 shows the cost for every started hour of EC2 instance depending on its type. As explained in test bed deployment the SaaS platform VMs were deployed into *m1.small* EC2 instances. The SCWA element of TBRAM system was deployed into *m1.medium* instance as an equivalent of the Base System's ELB. The ELB is a special EC2 instance type and it is a part of AWS EC2 services. Table 2 in turn shows the cost per GB of data transferred through AWS. Normally we pay only for data transferred out of the cloud environment. In case of the ELB there is small difference, we pay also for data transferred in/out of the ELB (even when deployed into the same Availability Zone of a region) as well as for each GB processed by it. After this short introduction to payment details of AWS it is the

time to present the actual economical difference between the systems. In Table 3 EC2 cost of the differing parts of both systems is presented. One can notice that both workload types of tests were cheaper to conduct on TB RAM system with the overall cost reduction of 15.58%.

Table 3 EC2 cost difference between the systems

	Incremental	Peak-based	Total
Base System	13.40 USD	21.74 USD	35.14 USD
TBRAM	11.82 USD	17.85 USD	29.67 USD
Total	25,22 USD	39,59 USD	64,80 USD

5 Conclusions and Future Works

In this paper, cost-effectiveness was scrutinized from two different perspectives. First was based on server-hours number consumed by the systems during the test. The second one was based on the billing statement for AWS resources usage from Amazon. The TB RAM system used about 20% less server-hours in case of the incremental workload test and over 30% less in case of peak-based tests.

That system was also over 15% cheaper than the Base System. Therefore it can be said that the resource allocation based on tenants definitely influenced cost-effectiveness of the SaaS applications. So the answer for the first research question is positive. The second thing to examine was to check if and how the TB RAM improves a SaaS system. The results showed that this model statistically (with 97.5% accuracy) improved under-utilization of the cloud resources in case of peak-based workload. It is worth to add that other system characteristics were generally slightly improved or the same as for the Base System. According to that one can also state that the TB RAM improved the authorial SaaS system.

This research showed that TB RAM can improve the cost-effectiveness. However, this is just one side of a medal. Conformance to that model introduces non-negligible development overhead.

In the course of further research authors will focus on introduction of authorial load balancer into TB RAM architecture to compare results from a SaaS system based on ELB in terms of tenant-based resource scaling.

It is expected that dynamic resource scaling based on authorial approach significantly reduce server-hours.

References

- [1] Amazon Auto Scaling, <http://aws.amazon.com/autoscaling/> (accessed: December 07, 2012)
- [2] Architecture Strategies for Catching the Long Tail (2006), <http://msdn.microsoft.com/enus/library/aa479069.aspx> (accessed: December 07, 2012)

- [3] Armbrust, M., et al.: Above the Clouds: A Berkeley View of Cloud Computing. Technical Report #UCB/EECS-2009-28. Electrical Engineering and Computer Sciences University of California at Berkeley (2009)
- [4] Bientinesi, P., et al.: HPC on Competitive Cloud Resources. In: Furht, B., Escalante, A. (eds.) Handbook of Cloud Computing, pp. 493–516. Springer US (2010)
- [5] Guo, C.J., et al.: A framework for native multi-tenancy application development and management. In: 2007 9th IEEE International Conference on e-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, e-Commerce, and e-Services, Piscataway, NJ, USA, July 23-26, pp. 470–477 (2007)
- [6] Chen, Y., et al.: An Efficient Resource Management System for On-Line Virtual Cluster Provision. In: IEEE International Conference on Cloud Computing, CLOUD 2009, pp. 72–79 (September 2009)
- [7] Dyachuk, D., Deters, R.: A solution to resource underutilization for web services hosted in the cloud. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2009, Part I. LNCS, vol. 5870, pp. 567–584. Springer, Heidelberg (2009)
- [8] Espadas, J., et al.: A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures (2011)
- [9] Cai, H., et al.: An end-to-end methodology and toolkit for fine granularity SaaS-ization. In: 2009 IEEE International Conference on Cloud Computing (CLOUD), Piscataway, NJ, USA, September 21-25, pp. 101–108 (2009)
- [10] Iyer, R., et al.: VM3: Measuring, modeling and managing VM shared resources. *Computer Networks* 53(17), 2873–2887 (2009)
- [11] Mc Evoy, G.V., Schulze, B.: Using clouds to address grid limitations. In: 6th International Workshop on Middleware for Grid Computing, MGC 2008, held at the ACM/IFIP/USENIX 9th International Middleware Conference, Leuven, Belgium (2008)
- [12] Meng, X., et al.: Efficient resource provisioning in compute clouds via VM multiplexing. In: 7th IEEE/ACM International Conference on Autonomic Computing and Communications, ICAC 2010, Washington, DC, United states, pp. 11–20 (2010)
- [13] Mishra, A.K., et al.: Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters. *Performance Evaluation Review* 37(4), 34–41 (2010)
- [14] Multi-Tenant Data Architecture (2006), <http://msdn.microsoft.com/en-us/library/aa479086.aspx> (accessed: September 07, 2012)
- [15] Paroux, G., et al.: A Java CPU calibration tool for load balancing in distributed applications. In: Proceedings - ISPDC 2004: Third International Symposium on Parallel and Distributed Computing/HeteroPar 2004: Third International Workshop on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks, Cork, Ireland, pp. 155–159 (2004)
- [16] SaaS Capacity Planning: Transaction Cost Analysis Revisited (2008), <http://msdn.microsoft.com/en-us/library/cc261632.aspx> (accessed: September 07, 2012)
- [17] Stillwell, M., et al.: Resource allocation algorithms for virtualized service hosting platforms. *Journal of Parallel and Distributed Computing* 70(9), 962–974 (2010)
- [18] Wee, S., Liu, H.: Client-side load balancer using cloud. In: 25th Annual ACM Symposium on Applied Computing, SAC 2010, Sierre, Switzerland, pp. 399–405 (2010)
- [19] Wu, Q., Wang, Y.: Performance testing and optimization of J2EE-based web applications. In: 2nd International Workshop on Education Technology and Computer Science, ETCS 2010, Wuhan, Hubei, China, March 6-7, pp. 681–683 (2010)
- [20] Yang, J., et al.: A profile-based approach to just-in-time scalability for cloud applications. In: CLOUD 2009 - 2009 IEEE International Conference on Cloud Computing, Bangalore, India, September 21-25, pp. 9–16 (2009)

Automatic Load Testing of Web Application in SaaS Model

Emil Stupiec and Tomasz Walkowiak

Wrocław University of Technology, Wybrzeże Wyspiańskiego 27,
50-320 Wrocław, Poland
tomasz.walkowiak@pwr.wroc.pl

Abstract. Necessity of monitoring in combination with the actual complexity of the e-services creates a need for constructing systems for active monitoring of various types of web services. Usually those systems are high-availability services, that require on one hand ingenious software solutions and on the other hand reliable hardware architecture. The created systems need to be flexible enough to satisfy customers requirements. This paper introduces an example solution of a system, that implement functional monitor of services provided in SaaS model. The provided system allows to check certain functionalities or whole service by running functional/load tests scenarios that are automatically generated, based on specially prepared user model.

1 Introduction

The great and growing access to global networks results in the great popularity and ubiquity of web applications. One of the principles in serving internet applications is a need to deal with a large number of customers. Moreover, those applications have to be fast and reliable, as well as up-to-date. Of course one can meet different kind of problems of using the Internet as common as delays during its usage which are in interest of different studies [4], [5]. There can be noticed rapid change of web sites due the different factors such as visibility in search engines or other web pages. These demands for implement practices that make sure the web-site will meet the requirements as well as for optimization of its variety of components [3][7]. One of these practices is the load testing [3][12],[15]. This is one of the more important method in ensuring the above demands. Growth of these kinds of interfaces, accessed through the users request, can be adjusted to the similar usage patterns as web interfaces.

Therefore, from very beginning there was a need to develop systems that perform web servers benchmarking and represents different approaches to defined problem [1], in order to automate the process of load testing. In this field one can find various solutions such as Funkload [6], Apache JMeter [2], Rational Performance Tester [10] or Developer Tools [16] from Microsoft. Base drawbacks of those tools are the poor and very simplified testing scenario (usually not based on

any reliable data, they are created “by hand”). What is more, the fact, that those tools are single applications that are usually running on a single computer what leads to a great testing limitation. This is a huge drawback, because it can run the load test for relatively low number of parallel scenarios. When it comes to test the servers that need to serve thousands or more users (so load test need to be run at assumed capacity threshold level), there can be a problem with providing the sufficient hardware and network bandwidth resource. That’s why authors propose to use cloud computing model called Software as a Service (SaaS) [14].

The paper is organize as follows, next chapter defines problems of load testing. Then in chapter 3 a workload model is presented. Next chapter analyses the problem of using session and form data. It is followed be a description of proposed architecture of load test system working in a SaaS model. Finally the paper is concluded by a short critical analysis of the proposed approach.

2 Problem Analysis

Developing useful tool for load testing of web applications requires solving following problems:

- how to prepare a reliable, realistic workload,
- how to collect and what data is necessary for tests,
- how to ensure sufficient hardware infrastructure for load testing,
- how to automate (and in what automation level) the processes in solution, to get easy to use and flexible system.

The test model provided in load test needs to be as realistic as it can be [3][12]. It is a crucial foundation in this manner. The load test validity is dependent mainly on simulating realistic behavior, because not providing realistic user model, can produce inconsistent results [15].

The second problem is a base on which the future tests are built. It requires collecting data that are used by the web application users. An important aspect are session data, which can be mostly gained by logging web traffic. This type of data is potentially meaningful and need to be properly analyzed and interpreted. This process of giving the sense out of Web-logging data is called Web mining [11].

The third problem, sufficient hardware infrastructure, is often quite underestimated and it may lead unwittingly to inconsistent results. One of solution would be to provide some kind of multiagent system or such solution as presented in [9]. They should scatter equally as the web applications’ users and should allow to run created test separately (or divide the load of the tests’ units to each of the agents). However, it is difficult to develop due to financial costs, especially in respect to efficiency. This is because load testing is computationally demanding and require dedicated hardware which on the other hand can be idle in most of the time (one don’t run load test all the time). Nevertheless nowadays one could use the new coming infrastructures that may provide a solution to that problem, i.e. cloud computing.

The last problem is how to automate the whole process of load testing from the very beginning to the very end and the detail level (some parts of the system may not be needed to be automated). Generally the log analysis and interpretation and model generation (the two basic system modules) should be processed fully automatically. Such approach could give a very comfortable solution that saves a lot of testers time.

We propose to solve most of mentioned problem using stochastic workload model (presented in the next chapter), gaining model parameters (mainly session and forms data) from web log analysis (the problem is presented in chapter 4) and effective usage of computational power within SaaS cloud computing model in Amazon EC2 infrastructure (chapter 5).

3 The Workload Model

We propose to build the user model based on the stochastic workload model proposed in [13]. It applies to the interesting methodology of form-oriented analysis [8]. This methodology is based on building submit-response style models as a bipartite state diagram (decision three). Additionally the model is extended with the stochastic functions that describe very important, from point of view of realism, aspects like navigation, time delays and user input.

Form-oriented analysis [8] is a methodology for the specification of ultra-thin client based systems. Web application is described as typed, bipartite state machine which consists of:

- page - a set of screens and arbitrary number of forms,
- screen - a single instance of a particular page, as it is seen by user in the web browser; one page may have similar look but with content changes for different types of user,
- forms that contains arbitrary number of fields and a way to submit the fields input information,
- fields allow users enter some information,
- submission invokes an action on server side,
- action - processes the submitted information and returns new screen in response,
- hyperlinks - forms without fields (or hidden to the user).

One can visualize the models as a formchart, like presented in Fig. 1. In a formchart pages are represented as ovals, action as boxes and transitions between previous two as arrows. They are forming a directed graph as a whole. The Fig. 1 shows only a part of some bigger system. Thanks to the formcharts one can represent not only the whole system, but also select some of the functionalities/parts of the system and represent (consequently test) them.

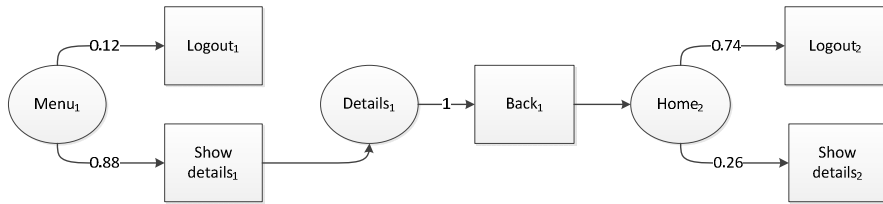


Fig. 1 Sample formchart

As the formchart can specify possible behavior due to its specification, one can model what are the next action invocations from those, which are available for the user in current web application context. The navigational choices have assigned stochastic values as is shown on the picture (numeric values on transitions). Of course the behavior of server is deterministic so there is no need to set any stochastic values on server action. Additionally the message model can be implemented (for filling the form data), as for example pair values persistent in fast non-SQL databases (seems like most efficient way for large amount of input data from logs). Also for implementing the realistic user acting scenario, there need to be taken under consideration the timing of user behavior. In web applications it boils down to provide a so called “think time” value. As there will be access to session data this value can be extracted from there.

Those from charts, as it has been said, will be featuring the history sensitivity. These is, because when building an empirical model based on session data, one can capture difference between the users’ decision making on one page in different cycles of session lifetime. In example when user logs in into the web applications’ home page he is less likely to logout, than doing any other activities, but when some activities are performed and he will come back to the home page there is much bigger probability that he will logout. In this case there will be build a decision tree, that in different usage cycles, will give to the repeating pages incremental index (as it is presented exactly on Fig. 1). This stochastic formcharts is similar to the Markov chains. There is of course important difference between them. The Markov chains create state machine with probabilities on every transitions, formcharts on the other hand are bipartite state machines with probabilities only on transitions from pages to actions. What is more the solution captures behavior over time.

Building a stochastic model is quite easier than deterministic ones, where deterministic test means that the transitions between certain actions are going to occur or not (0 or 1), for example those that uses prerecorded user’s session. Maybe one could not notice this in simple, small webpage, but large internet applications will show that the amount of tests will be significant, with only testing the most crucial functionalities. One need remember that possible sequences grows exponentially with greater number of actions.

The second thing is that when using stochastic approach, one need to realize that there may occur every possible sequence of actions, which may produce effects that remain hidden when using prerecorded sessions. An example would be,

actions in a system may compete for the same resources, such as memory, processor time or database locks [13]. Longer the stochastic load tests are, greater the probability of such situation occurrence is. Finally it allows to find critical scenarios a-posteriori by analyzing collected data.

4 Setting Model Parameters

In a manner of specifying the load tests itself the first thing is to set appropriate parameters to forms. It has no sense to provide the insensible data there. To provide the appropriate data to fill the forms in automatic tests, this data need to be collected during the log generation. Unfortunately there is no readymade mechanism to collect it. Normally, it is not important in logging and even more, it is not good to store it in non-protected persistent files. Therefore, the mechanism (it can be implemented various ways - using JavaScript, adding a processing resource into internet application as i.e. filters) of form data collection need to be added to web application and stored data should be protected from unauthorized read. This provided data will be used then to fill the forms in web application. The usage will reflect exact participation of provided data in selected form, like in Fig. 2.

```
form1 = {  
  24, login: janK, password: kowal;  
    10, login: KJan, password: janek;  
    31, login: JohnK, password: kolek;  
    5, login: KOWJ, password: johnny;  
    30, login: janek1234, password: 1234;  
}
```

Fig. 2 Sample form data

Where first argument is the form name, by which right form will be identified. Then we've got at least two comma separated values (first show how many times data has occurred in provided logs, the second and so on, because there can be more than one input field in form, have pair names of the input). One can interpret the example data as that there will be respectively 0.24, 0.1, 0.31, 0.5, 0.3 probabilities of occurring the provided data in the run test. In this approach for providing data there is no need to take into consideration manner of inadequate/mistaken input values. If they occur, they will be taken into the consideration. This is a little different approach to [13] as this will always be based on real session data not on top-down estimation. In [13] approach there is a fixed probability of providing wrong credentials. When the described situation occurs, then the random credentials are given as an input.

Also worth of mentioning is the situation of session dropouts, i.e. a finish of the session on web page by user. It appears due to closing of the web browser,

browsing the web page out of the application or simply session timeout. First thing is to model this situation. This is done by adding artificial action that for example can be called “Terminate”. As it can be reached from every web page, this action will be connected by transitions from all others actions. Next thing is to assume the moment when it happens. Unfortunately a lot of web applications do not check if it occurs. Then to make possible to model it, during the session data analysis, we need to be set a maximum acceptable “think” time. The best solution would be to set this time equal to the servers’ session dropout (then anyway session is finished and when user will come back, he need to log in again). Of course then in user model “think time” in this last transition is set to 0 and go straight to the “Terminate” action.

All data, that are provided for setting up model parameters, are based on real user session data. If there is a need to provide parameters data in newly created web application a good solution is proposed in [12] and [13]. Whereas from the perspective of monitoring the internet application, it is already running software, whence the data can be extracted from its actual usage. Collecting session data is not an easy task. An overview of data analysis method is provided in [11] and [17]. One need to watch out not to choose too few logger results and also make data too redundant (of course second situation is less harm in overall).

We propose to focus of following data:

- session (user) distinction,
- action type,
- form parameters,
- “think time” values,
- time results for final load tests results.

As it has been said not all data can be provided straight from the provided logs. For example to provide form data for POST methods, there is a need to add such mechanism to the web application, that will add these fields to log. The logs from various web servers are different and due to fact that they support different programming languages the mechanisms for log gathering may be different (but not necessarily) and log analyzer is usually a “spare part” (one need to change it when the web server will change).

5 Compute Cloud

Nowadays the still growing need for providing more and more powerful hardware for demanding software and on the other hand the need to reduce all types of cost, leads to development of new computing solutions. The autonomous machines (PC’s) are reaching their limit in computational power and new solutions in this manner may not seem to be cost effective. Fortunately now one can get an easy and comfortable access to high computational potential thanks to a relatively new

approach to this issue, the cloud computing. This approach is a connection of the old and well known techniques (referring for example to the mainframe model) and the notion of virtualization. Thanks to that one can get access, even in perspective of quite demanding software, to unlimited computational potential in lot of convenient configurations (by i.e. virtualizing one very powerful machine or several weaker). The approach proposed in this paper uses cloud computing as an infrastructure for serving all parts of the load testing system.

We have developed the load testing application on Elastic Compute Cloud (EC2) from Amazon Web Services (AWS). It handles the interaction with the compute cloud, which includes handling the user's tests, storing and running them in multiple virtual machines. Moreover, it includes manager of those load test including the simple thin client interface for managing those load test. The preliminary tests shows the usefulness of implementing such system in compute cloud. We managed to deal with 3800 concurrent clients (on 19 virtual machines). It shows that using cloud computing allows to web application with real large number of clients. Based on this experience, we propose to build the load test system as presented on Fig.3. Following systems elements can be distinguished:

- web server - providing expanded user interface for managing/running tests and gather the results for registered users,
- testing server - handling communication and managing the compute cloud, providing communication between all parts of the system, running test instances and gathering data,
- log analyzer - analyses the log and produces as an output data for the test,
- test application - program that based on session data perform stochastic evaluation and runs the scenario of the test,
- test trigger - application, which is running on each virtual machine, that is running tests; it's task is to prepare the test environment and run the test.

The scenario of using this system is: the registered user creates "project" (testing request for certain web application) and creates simulation (single scenario for test). After creating a project the log file is provided at any time (automatically or by hand), when that happens the log analysis is automatically triggered by the testing server. When input data for test is ready the load test is launched at any given time. The analysis can be run also at any given time not interfering with produced analysis. One has to specify the number of the users for the load scenario for calculation how many instances has to be started.

The instances are launched and prepared for the test. When they are ready, the synchronized load test is being launched on the virtual machines. After the test, the results are provided and different specifics can be seen on the user web interface. As it can be noticed the system works almost automatically. There are a few parameters of the scenario that has to be set by hand, but they are necessary from the perspective of user-system interaction and helps better test cases handling.

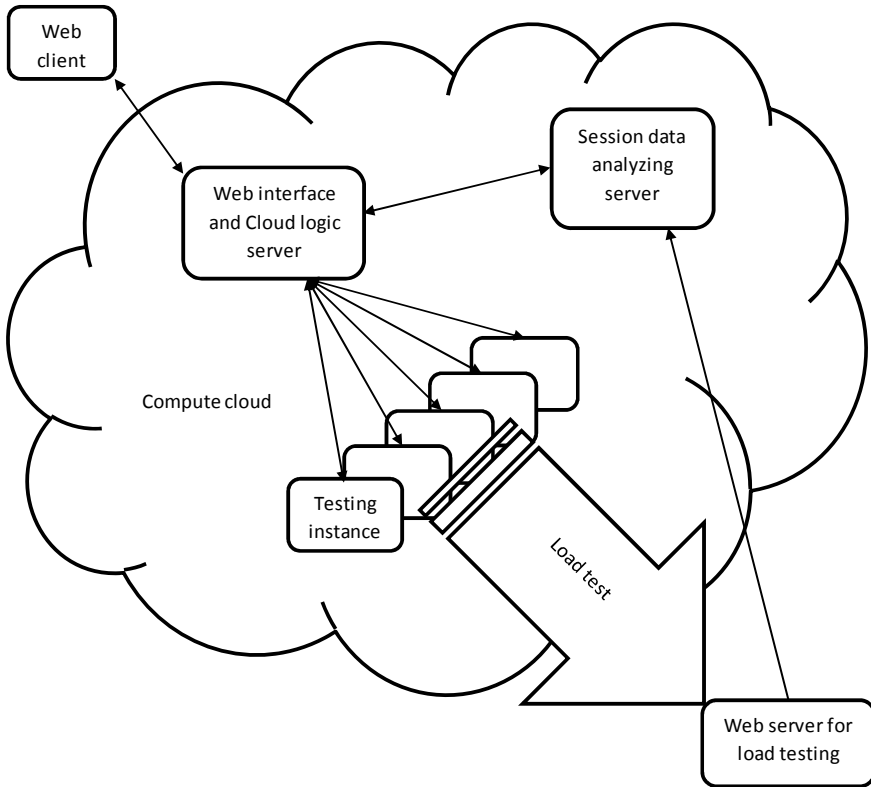


Fig. 3 Load test system structure

6 Conclusions

We have shown a proposition of the realistic and effective web load test system. The proposed system allows to check certain functionalities or whole service by running functional/load tests scenarios that are automatically generated, based on specially prepared user model.

The system realism is achieved by a usage of the form-oriented stochastic user model (slightly modified model from [13]). The model includes such elements as stochastic navigation of a user, based on session history, stochastic parameter generation for forms, and right workload parameters (defined as a input data for tests). Of course looking from the perspective of stochastic specification, it not give guarantee of true realism, but a substitution that is actually the best approach in the field of study. Also the best way to gather data would be approach that not impact in the web application solution, but now it seems that web servers don't give possibility to overcome this issue. But on the other hand the idea of the

separate system for load testing seems like good solution as it is quite important aspect in developing and monitoring web applications. Providing solution as here also give access for different type of users in one place and dedicated systems can be easily developed, and for example subject to the popular trend of generally conceived notion of socializing it. The adapted methodology of submit-response style systems built can also be implemented in any web services too.

The aspect of the effectiveness of the proposed approach was achieved as a result of decision of implementing the load system in SaaS model on Amazon EC2 infrastructure. Fundamental aspect of this decision lays in a usage of the computing utility in efficient way to handle this problem. The Amazon EC2 gives the scalable computational power, that can be used and set up to the current infrastructure user needs. In traditional way the maximal needed computational potential had to be provided at very beginning (additional problem with estimation for how much is required) to handle the load tests. The other important aspects are financial costs. Current costs¹ (March 2013) per instance-hour are from 0.02\$ for minimal (Micro) instance to 4.60\$ for maximal (Eight Extra Large) instance. If there are instances that are planned to be running all the time, it can be reserved, which can reduce the costs even more. Then one pays constant month upfront payment during the reservation time. The additional costs are less important because they are relatively small in example I/O request are counted by a cents per million, data transfer in cents per GB, cloud localization differences in even less than a cent.

References

- [1] Mark, A.: Performance test tools (2013), <http://www.opensourcetesting.org/performance.php> (accessed March 28, 2013)
- [2] Apache Software Foundation. Apache JMeter vers. 2.7 (2012), <http://jmeter.apache.org> (accessed March 24, 2013)
- [3] Banga, G., Druschel, P.: Measuring the Capacity of a Web Server under Realistic Loads. *World Wide Web Journal* 2, 69–83 (1999)
- [4] Barford, P., Crovella, M.: Measuring Web Performance in the Wide Area. *ACM Perform. Eval. Rev.* 27, 37–48 (1999)
- [5] Curran, K., Duffy, C.: Understanding and Reducing Web. *Int. J. Netw. Manag.* 15, 89–102 (2005)
- [6] Delbosc, B.: Funkload documentation (2011), <http://funkload.nuxeo.org> (accessed April 3, 2013)
- [7] Draheim, D., Grundy, J., Hosking, J., Lutteroth, C., Weber, G.: Realistic load testing of web applications. In: *CSMR 2006: Proceedings of the 10th European Conference on Software Maintenance and Reengineering* (2006)
- [8] Draheim, D., Weber, G.: *Form-Oriented Analysis - A New Methodology to Model Form-Based Applications*. Springer, IEEE Press (2004)

¹ <http://aws.amazon.com/ec2/instance-types/>

- [9] Huiming, Y., Jin, Z., Jinsheng, X.: Multi-Agent Testing Architecture for Monitoring and Analyzing the Performance of Web Applications. In: 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse, pp. 115–120 (2006)
- [10] IBM. Rational Performance Tester (2011), <http://www142.ibm.com/software/products/us/en/performance> (accessed March 24, 2013)
- [11] Juvina, I.: Development of a Cognitive Model for Navigating on the Web. Dissertation, Utrecht University (2006)
- [12] Lei, X., Baowen, X., Jixiang, J.: Testing Web Applications Focusing on their Specialties. SIGSOFT Softw. Eng. Notes 30, 10–16 (2005)
- [13] Lutteroth, C., Weber, G.: Modeling a Realistic Workload for Performance Testing. In: Enterprise Distributed Object Computing Conference, pp. 149–158 (2008)
- [14] Mäkilä, T., Järvi, A., Rönkkö, M., Nissilä, J.: How to Define Software-as-a-Service – An Empirical Study of Finnish SaaS Providers. Software Business 51, 115–124 (2010)
- [15] Menascé, D.A.: Load Testing of Web Sites. IEEE Internet Computing 6(4), 70–74 (2002)
- [16] Microsoft. Developer Tools (2012), <http://msdn.microsoft.com/enus/library/vstudio/dd293540.aspx> (accessed March 24, 2013)
- [17] Taksa, I., Spink, A., Jansen, J.B.: Web Log Analysis: Diversity of Research Methodologies. In: Handbook of Research on Web Log Analysis (2008), doi:10.4018/978-1-59904-974-8.ch025

Implementing Salsa20 vs. AES and Serpent Ciphers in Popular-Grade FPGA Devices

Jarosław Sugier

Wrocław University of Technology
Institute of Computer Engineering, Control and Robotics
ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland
jaroslaw.sugier@pwr.wroc.pl

Abstract. Salsa20 is a 256-bit stream cipher that has been proposed to eSTREAM, ECRYPT Stream Cipher Project, and is considered to be one of the most secure and relatively fastest proposals. This paper discusses hardware implementations of this cipher in two organizations – a fully unrolled, pipelined dataflow path and an iterative loop – in low-cost Field Programmable Gate Arrays, and compares them with equivalent realizations of the AES and Serpent block ciphers. The results demonstrate potential of the algorithm when it is implemented in the specific FPGA environment and evaluate its effectiveness in contemporary popular programmable devices.

1 Introduction

Efficient hardware implementation is important for any cipher. In this paper we evaluate potential of Salsa20 – a relatively novel cipher known from eSTREAM project – in FPGA implementations comparing it with the two older, more established AES and Serpent algorithms.

In the next section the Salsa20 cipher is briefly presented (AES and Serpent are not covered but the reader can find information about them in the specifications [6] and [1] as well as in our previous work [8], for example). Then, in section 3, the two basic architectures for cipher organizations are introduced and in section 4 the results obtained after FPGA implementation are discussed.

2 The Salsa20 Cipher

Salsa20 family of stream ciphers [2]-[3] has been developed in 2005 by Daniel J. Bernstein from the University of Illinois at Chicago, USA, and submitted to the eSTREAM project. After passing all phases of selection unmodified it has been included in the final portfolio of Profile 1 (software) ciphers along with 4 other proposals.

At its core the method is a 64B[yte] hash function which operates in the counter mode as a stream cipher. The state is also 64B large and is represented as a series

of 32b[it] state words $\mathbf{q} = (q_0, q_0, \dots, q_{15})$. In contrast to block ciphers like AES and Serpent, there is no separate key expansion path running in parallel with data path which would compute a separate key for each round; instead, the secret external key is directly entered into the input 64 bytes and then transformed in the cascade of rounds like other input data.

The hash function consists in a series of 20 rounds which are executed over the state \mathbf{q} . The elementary organizational unit is a *quarterround* function which transforms four state words: $quarterround(w_0, w_1, w_2, w_3) = (w_0', w_1', w_2', w_3')$ such that

$$\begin{aligned}w_1' &= w_1 \oplus ((w_0 + w_3) \ll 7) \\w_2' &= w_2 \oplus ((w_1' + w_0) \ll 9) \\w_3' &= w_3 \oplus ((w_2' + w_1') \ll 13) \\w_0' &= w_0 \oplus ((w_3' + w_2') \ll 18)\end{aligned}$$

where ' \oplus ' denotes bitwise XOR operation, ' $\ll n$ ' stands for left rotation by the n bits and '+' sign here represents addition modulo 2^{32} . The flow of data which results from the above equations is graphically visualized in the left part of Figure 1.

Four quarterrounds operating in parallel transform the entire state vector \mathbf{q} into \mathbf{q}' and constitute a complete round of the cipher. When numbering the rounds from 0 to 19, the even numbered ones are called *column rounds*:

$$\begin{aligned}(q_0', q_4', q_8', q_{12}') &= quarterround(q_0, q_4, q_8, q_{12}) \\(q_5', q_9', q_{13}', q_1') &= quarterround(q_5, q_9, q_{13}, q_1) \\(q_{10}', q_{14}', q_2', q_6') &= quarterround(q_{10}, q_{14}, q_2, q_6) \\(q_{15}', q_3', q_7', q_{11}') &= quarterround(q_{15}, q_3, q_7, q_{11})\end{aligned}$$

while odd numbered ones are called *row rounds*:

$$\begin{aligned}(q_0', q_1', q_2', q_3') &= quarterround(q_0, q_1, q_2, q_3) \\(q_5', q_6', q_7', q_4') &= quarterround(q_5, q_6, q_7, q_4) \\(q_{10}', q_{11}', q_8', q_9') &= quarterround(q_{10}, q_{11}, q_8, q_9) \\(q_{15}', q_{12}', q_{13}', q_{14}') &= quarterround(q_{15}, q_{12}, q_{13}, q_{14})\end{aligned}$$

with the only difference being in permutations of the q_i words at their inputs. A column round followed by a row round make up a *double round*:

$$doubleround(\mathbf{q}) = rowround(columnround(\mathbf{q}))$$

To generate the hash value for a 64B input \mathbf{x} , first the double round is applied ten times and then the result is added:

$$Salsa20(\mathbf{x}) = doubleround^{10}(\mathbf{x}) + \mathbf{x}$$

(strictly speaking, since all the transformations are defined for a sequence of 4B words, if \mathbf{x} in the above equation is to be a byte stream, it will need to be transformed using little endian notation).

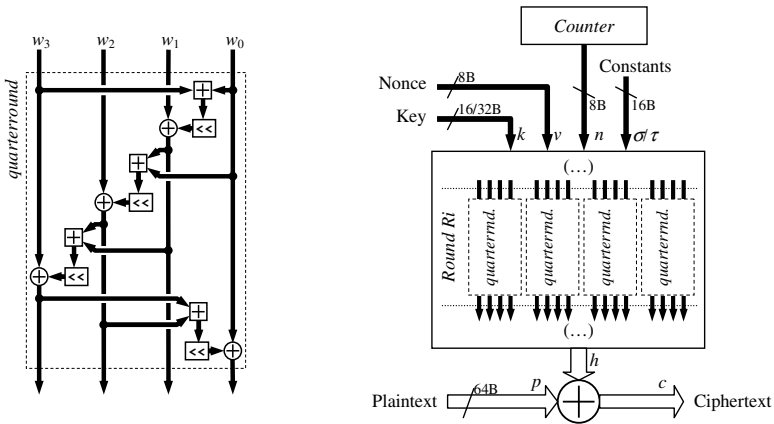


Fig. 1 The Salsa20 single quarterround transformation (left) and the complete hash function working as a cipher module in the counter mode (right)

Operation of the hash function as a stream cipher in the counter mode is shown in the right part of Figure 1. The input is made from 32B of the *key* (which is split into the two halves k_L and k_H) with 8B *nonce* v (*number once* - a unique message identifier) plus 8B *counter* n and 16 *constants bytes* σ_i :

$$h = Salsa20(\sigma_0, k_L, \sigma_1, v, n, \sigma_2, k_H, \sigma_3)$$

where the constants $(\sigma_0, \sigma_1, \sigma_2, \sigma_3)$ make up an ASCII encoded string “expand 32-byte key”. In this paper we will work with the version of the algorithm when the key is 16B – in such a case it is just repeated twice in the input in place of k_L and k_H and slightly different constants are applied.

During encryption the result h is XOR’ed with the plaintext; during decryption, the same hash result is XOR’ed with the ciphertext, hence the same module can be used. Since the counter n (which is incremented for each block of the generated cipher) is 64b long, the maximum length of the encoded stream is limited to 2^{64} 64B blocks or 2^{70} bytes (approx. 1 billion TB).

3 Architectures for Hardware Implementations

In this study we evaluate efficiency of hardware implementation of the three ciphers for the two basic types of processing: pipelined dataflow and iterative organizations which are introduced in the following two subsections.

3.1 Pipelined Architectures

These architectures were developed as a pipelined modification of a strictly combi-national implementation of the ciphers. First, all the rounds (20 for Salsa20, 11 for AES or 32 for Serpent) were implemented as separate hardware modules that

created a continuous path from the input to the output registers. In-between, the designs operated as combinational functions that mapped 512 (or 256) input bits (data + key) into 512 (or 128) output bits of the hash or ciphertext. The designs, in all three cases, were specified by translating as closely as possible the original specifications into the VHDL language using strict RTL style. In the cases of AES and Serpent the substitution boxes, both 8b- (AES) and 4b-wide (Serpent), were defined as in generic Xilinx templates that are recommended for ROM specification.

Such a combinational sequence of the cipher rounds was converted into a cascade of pipeline stages by registering the state signals at the round boundaries with the key expansion path remaining combinational. The complete pipeline had 20 (Salsa20), 11 (AES) or 32 (Serpent) stages.

3.2 Iterative Architectures

The iterative architectures proposed in this study are based on the structure of one stage taken from the pipelined architectures which was supplemented with necessary multiplexing logic (loading the data in – looping back – loading the data out) and a simple controller responsible for counting the repetitions of the loop (round numbers) and supervising the multiplexers. The controller comprised just a single “idle/busy” register plus a rudimentary counter and such a minimal form proved to be sufficient in all the three cases.

This simple concept was slightly modified in case of the Salsa20 cipher: since the double round, i.e. the conjunction of a column round followed by a row round, is the unit being repeated 10 times in the whole cipher path, it was selected to use such a block and not a single round as the contents of the iteration loop. This meant that the size of the hardware needed to be doubled (two rounds implemented instead of one) but, since the individual Salsa20 rounds are not strictly identical using different permutations of the input state words, taking one round would need additional multiplexers for differentiating odd and even iterations. As it was proved in [9] extra cost of such multiplexers would significantly reduce the savings in design size and would harm the performance.

Table 1 Implementation results for the pipelined architectures

	Spartan-6			Spartan-3		
	Salsa20	AES	Serpent	Salsa20	AES	Serpent
min T_{clk} [ns]	15.2	5.13	5.10	24.0	12.0	7.00
f_{max} [MHz]	65.9	195	196	41.6	83.5	143
Latency [T_{clk}]	20	11	32	20	11	32
Latency [ns]	303	56.4	163	481	132	224
Throughput [Gbps]	33.7	24.4	24.5	21.3	10.4	17.9
Mbps / Slice	6.36	9.87	5.46	1.74	0.57	1.55
Slices	5 307	2 529	4 590	12 254	18 799	11 793
Slice LUTs	21 140	9 087	15 523	21 235	30 426	22 708
Registers	11 008	1 536	4 224	11 008	5 061	4 224

Table 2 Implementation results for the iterative architectures

	Spartan-6			Spartan-3		
	Salsa20	AES	Serpent	Salsa20	AES	Serpent
min T_{clk} [ns]	20.8	6.26	5.57	51.7	13.0	10.4
f_{max} [MHz]	48.0	160	180	19.4	77.0	96.2
Latency [T_{clk}]	10	11	34	10	11	34
Latency [ns]	208	68.9	189	517	143	353
Throughput [Gbps]	2.46	1.81	0.66	0.991	0.875	0.354
Mbps / Slice	3.00	3.77	1.26	0.487	0.15	0.17
Slices	818	493	536	2 036	5 948	2 145
Slice LUTs	2 955	1 367	1 566	3 374	7 986	3 995
Registers	1 317	817	806	1 286	781	783

Thus, in these organizations one block of data was encoded in 10 clock cycles for Salsa20, in 11 clock cycles for AES and 34 clock cycles for Serpent (in this particular cipher it was needed to insert two extra cycles for preparation of round keys [7]).

4 Implementation Results

As the implementation hardware platform it was chosen to test two popular-grade families of FPGA devices from Xilinx, the inventor and still one of the most successful suppliers of the programmable logic: an older, now more archetypal, Spartan-3 [10] and a newer Spartan-6 [11]. The results for AES and Serpent are taken from our previous work in [8] where these designs followed the same methodology and were verified in identical FPGA chips. The Salsa20 implementations presented here are the original contribution and their evaluation against the two older ciphers is the point of main interest in this work.

In total, there are 6 designs in the comparison (2 architectures for each cipher) and *the same* code was implemented in Xilinx ISE Design Suite version 14.3 twice: for Spartan-6 (XC6SLX150) and Spartan-3 (XC3S2000) devices. Implementation was fully automatic, without any hand-made fine tuning neither in placement nor in routing.

For every design we present the basic speed and size parameters: the minimum clock period (T_{clk}) and the maximum operating frequency (f_{max}) as they were estimated by the post-place & route static timing analysis, the latency expressed in clock cycles and in nanoseconds, the overall throughput in Gbps calculated for the f_{max} , size of the design expressed in the number of occupied slices, LUTs and registers, and a synthetic performance measure which is commonly used for estimation of speed vs. size efficiency – Mbps of the throughput per one occupied slice.

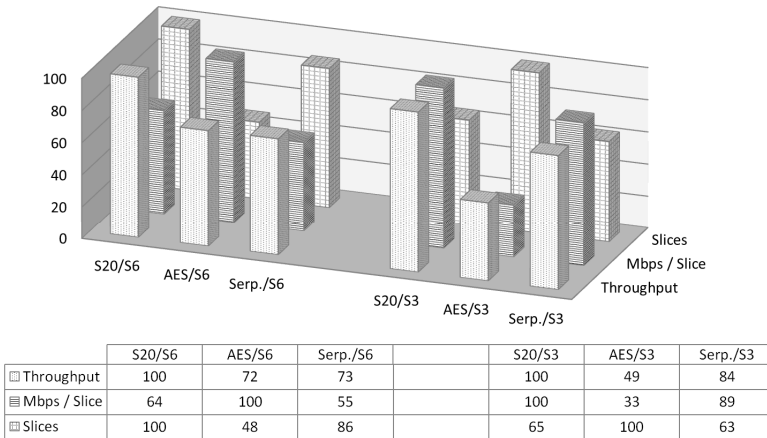


Fig. 2 Relative measures for the pipelined architectures (max value per device = 100)

The results of the pipelined architectures are shown in Table 1 and, first of all, they confirm superior throughput levels that can be reached when the completely unrolled iterative loops of the ciphers are converted into the pipelines: the designs reach from 24 to 34 Gbps in Spartan-6 and from 10 to 21 Gbps in Spartan-3 devices with the Salsa20 being the fastest algorithm in both cases. Comparing the latency parameter (which is not so clearly in favour of this cipher) it should be pointed out that it does not take into account different sizes of the processed block: while in Salsa20 with every clock tick the output produces 512b of the result, the AES and Serpent generate one quarter of this number, so (in the particular case of this organization) the 512 bits of data would occupy 4 consecutive pipeline stages and the actual latency would be higher.

The same remark must be made when comparing sizes of the designs: the output generated by the Salsa for the given numbers of the occupied slices is actually 4 times bigger than the ones generated by the AES and Serpent. On the other hand, the size of the AES in Spartan-3 is unexpectedly high because, as it was shown in [8], implementation of this particular cipher in the older architecture is problematic if no Block RAM resources can be additionally used.

The iterative architectures (Table 2) occupy from 5 to 9 times less slices than the corresponding pipelined ones (again with the exception of the AES implementation in Spartan-3) but, besides reasonably much lower throughput, their efficiency expressed in Mpbs per slice is 2 ÷ 4 times lower which means that the silicon area, although smaller, is not utilized as efficiently. Within this group of cipher architectures the Salsa algorithm again offers the highest raw throughput on both platforms but very small size of the AES in Spartan-6 makes this cipher the winner in the new FPGA family with regard to Mpbs / slice efficiency.

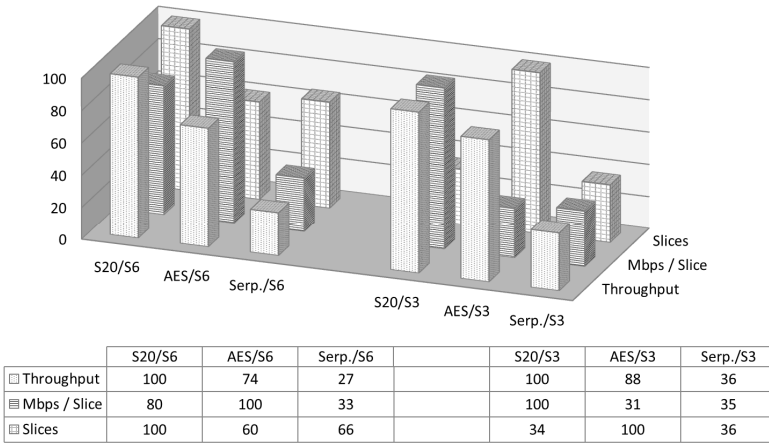


Fig. 3 Relative measures for the iterative architectures (max value per device = 100)

The three essential parameters of all the 12 implementations – the throughput, efficiency in Mbps / slice and size in slices – can be better evaluated using Figures 2 and 3 where they are measured against the maximum value for each device.

5 Conclusions

It is interesting to see effectiveness of the relatively new Salsa20 cipher against the older and more established contenders from the 1997 NIST competition: the AES and Serpent. Although actually a hash function at its core, the Salsa20 algorithm is also promoted as the more secure and the faster alternative to the AES standard. Even though it was selected to the final portfolio of eSTREAM project ciphers only in the software profile (and not in the hardware one), this work verifies Salsa20 potential when it is implemented in popular FPGA devices.

Despite fully automatic implementation and straightforward specification in the VHDL language, the cipher can reach a throughput levels over 20 Gbps in older Spartan-3 and over 30 Gbps in newer Spartan-6 devices when realized in the proposed pipelined architectures whereas the iterative organizations achieve, respectively, almost 1 and 2.5 Gbps. In all configurations the raw throughput of Salsa20 outperforms both the AES and Serpent in equivalent architectures. Other FPGA implementations of this cipher described in [4]-[5] and [12] are not directly comparable to the organizations proposed in this paper but have significantly lower speed (e.g. 1.2 Gbps in Spartan-3 device in [4]) and also lower throughput to area ratios (e.g. 0.74 Mbps/slice in [4] and 0.2 Mbps/slice in a highly iterative architecture proposed in [12]).

References

- [1] Anderson, R., Biham, E., Knudsen, L.: Serpent: A Proposal for the Advanced Encryption Standard. In: The First Advanced Encryption Standard (AES) Candidate Conference, Ventura, California, August 20-22 (1998), <http://www.cl.cam.ac.uk/~rja14/serpent.html> (accessed March 2012)
- [2] Bernstein, D.J.: The Salsa20 Stream Cipher. In: Proc. SKEW - Symmetric Key Encryption Workshop, Aarhus, Denmark, May 26-27, pp. 26–27 (2005), <http://cr.yp.to/snuffle.html> (accessed April 2013)
- [3] Bernstein, D.J.: The salsa20 family of stream ciphers. In: Robshaw, M., Billet, O. (eds.) *New Stream Cipher Designs*. LNCS, vol. 4986, pp. 84–97. Springer, Heidelberg (2008)
- [4] Gaj, K., Southern, G., Bachimanchi, R.: Comparison of hardware performance of selected Phase II eSTREAM candidates. In: Proc. State of the Art of Stream Ciphers Workshop, eSTREAM, ECRYPT Stream Cipher Project, vol. 26 (2007)
- [5] Good, T., Benaissa, M.: Hardware results for selected stream cipher candidates. In: Proc. State of the Art of Stream Ciphers Workshop, pp. 191–204 (2007)
- [6] National Institute of Standards and Technology, Specification for the ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197 (2001), <http://csrc.nist.gov/publications/PubsFIPS.html> (accessed March 2012)
- [7] Sugier, J.: Low-cost hardware implementation of Serpent cipher in programmable devices. In: *Monographs of System Dependability: Technical Approach to Dependability*, vol. 3, pp. 159–172. Publishing House of Wrocław University of Technology, Wrocław (2010)
- [8] Sugier, J.: Implementing AES and Serpent ciphers in new generation of low-cost FPGA devices. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability*. AISC, vol. 170, pp. 273–287. Springer, Heidelberg (2012)
- [9] Sugier, J.: Low-cost hardware implementations of Salsa20 stream cipher in programmable devices. *J. Polish Safety and Reliability Association* 4 (submitted for publication, 2013)
- [10] Xilinx, Inc., Spartan-3 Family Data Sheet. DS099.PDF (2009), <http://www.xilinx.com> (accessed April 2013)
- [11] Xilinx, Inc., Spartan-6 Family Overview. DS160.PDF (2011), <http://www.xilinx.com> (accessed April 2013)
- [12] Yan, J., Heys, H.M.: Hardware implementation of the Salsa20 and Phelix stream ciphers. In: Proc. Canadian Conf. Electrical and Computer Engineering, CCECE 2007, pp. 1125–1128. IEEE (2007)

On Testing Wireless Sensor Networks

Tomasz Surmacz, Bartosz Wojciechowski, Maciej Nikodem, and Mariusz Ślabicki

Institute of Computer Engineering, Control and Robotics,
Wrocław University of Technology
tomasz.surmacz@pwr.wroc.pl

Abstract. Testing wireless sensor networks (WSNs) is not a trivial task, due to the massively-parallel communication between many independently running processors. Distributed way of operation does not allow step-by-step execution and typical debugging, so other techniques have to be used, such as detailed packet logging and analysis. We provide a 2-way WSN-to-TCP proxy architecture which extends the typical BaseStation software with packet sniffing and packet sending capabilities. This allows writing and executing WSN test scenarios and automatic test assessment by using typical client-server applications written in any programming or scripting languages. An example of such protocol testing is also shown.

1 Introduction

The dominating application of Wireless Sensor Networks is environment monitoring. A typical setup used for monitoring consists of a WSN network, a data sink and possibly an off-site server that is used to collect, analyse and provide the data to involved parties. The WSN network in turn consists of a large number of nodes that are resource-constrained, i.e. have low performance, small storage capacity, and are usually battery-powered. The nodes in the network serve two purposes simultaneously – they collect data from sensors that they are equipped with and take part in routing the messages to the data-sink or Base Station (BS). Since nodes are equipped with low-power radio transceivers and small gain antennas (between 0 and 5 dBi), their effective communication range is usually very short. Therefore they usually communicate in a *multi-hop* fashion. Many layers of WSN-based monitoring systems make their behavior hard to understand, develop and test. It usually requires deep understanding of all levels, from sensor hardware and node software through network algorithms used in communication protocols stack, up to the system and application level. Even bridging the gap between network engineers and the experts in a given application requires a lot of effort [1].

In real conditions (out-of the lab) various problems may occur: hardware malfunctions, programming bugs or software incompatibility, when implicit

rules are not clear from the API of a module, or a layer of the software stack [5]. This necessitates strict testing of components of WSN networks on every level and performing tests of whole systems that are extensive both in time and network size. E.g. proper network behavior means also that the power consumption in different scenarios is bound by some limits and it takes time to test this. Due to presence of many physical effects on radio transmission, such as like fading, double- and multi-path reflections and interference, operating conditions of a WSN are hard to describe and simulate or recreate in laboratory. Therefore, creating realistic, controllable and repeatable tests is difficult [4].

Testing embedded software (e.g. for sensor nodes) poses another type of problems due to the lack of input/output devices or the possibilities of error logging. One option is to use JTAG interface to debug a single device, but such an approach is not an option when debugging distributed network algorithms. Debugging protocols is even harder, and non-scalable, as many devices would have to be run in testing mode simultaneously. Also, nodes usually communicate in event driven or asynchronous manner and any test should recreate and check time-dependencies.

There is no standard suite of benchmark applications for WSN-based systems, even though there were attempts at creating one. In [2] authors advocate creation of a standardized benchmark suite for TinyOS-compatible WSN nodes. They present sample benchmark results regarding performance and power consumption of different hardware components and call for further work in this area. Such standardized benchmark suite could be used to aid development of future hardware. Nazhandali et al. introduce new metrics in [7] that can be used to evaluate and compare wireless sensor systems as well as a set of applications representative of typical workload in WSNs. Predictions on how a large-scale WSN deployment will work can be made through simulations. This should provide an opportunity to eliminate some errors in early stages of implementation. However, in some cases even results of detailed simulations may stand in contradiction to real physical deployments [4] as it is difficult to simulate wave propagation and hence exact network behavior in sufficient detail.

In this work we describe our experiences with testing applications using TelosB nodes operating with TinyOS operating system [6]. Each node is equipped with 3 LEDs that can be used in simple debugging scenarios. E.g. to indicate sending or reception of a packet, sensor being turned on or a timer firing. However, these methods are useless for monitoring the whole network. Moving from simple experiments that test the properties of point-to-point communication [8] towards complete networks, results in increased software complexity. This increased complexity of a complete WSN testbed was previously described in literature. For example, the authors of [5] point to software complexity as one of the key reasons of not reaching the goals of their experiment. They also advocate applying rigorous software engineering approach to manage the growth in software complexity.

Operation of a WSN network is real-time in the sense that communication is usually asynchronous with time dependencies affecting the behavior of the whole network. Therefore it is not easy to test the proper operation of the whole network. One approach to this problem is to use traces of network communication that were gathered with a packet sniffer to make test scenarios. Then this recorded communication is injected into the network and new logs are analysed. This should result in much more accurate testing conditions.

To summarize, WSN testing can be done at different levels. At each level a different aspect is stressed:

1. node hardware – operation correctness in a range of physical environment conditions (e.g. temperature, humidity), performance, energy consumption with standardized benchmarks (including sleep modes),
2. node software – proper operation in response to any message (including erroneous and/or damaged packets),
3. network software – proper operation of the network protocol stack subject to proper and erroneous messages and varying amounts of communication,
4. system level – network and supporting infrastructure, including the BaseStation resiliency and correctness of the data analysis software.

2 Testing Environment Architecture

For the testing setup we have developed a sniffer-basestation architecture where an embedded system with a Linux operating system with a Wireless Sensor Network mote attached through a USB port acts as both the BaseStation and the network observation point. It can be a laptop computer running any available Linux distribution, or an embedded system, such as BeagleBoard or Raspberry Pi with its native Linux version.

The core of our architecture are two programs – TelosBaseStation and NetServ (Fig. 1). The first one is a TinyOS application running on CC2420-equipped motes, such as TelosB or XM1000 architectures. The BaseStation receives all the packets from the radio on a specified channel in a so-called promiscuous mode (i.e. sniffing all the packets that can be heard on the WSN network, not only those which were broadcast or addressed to the BaseStation). The received packets are timestamped and queued, then resent through the USB port. On the other end of this USB connection, the NetServ program receives these packets and makes them available in various ways, by storing them locally (acting as the packet sink) but also acting as a TCP proxy, by offering them to clients in many formats. Multiple clients can connect simultaneously to get these messages either in binary form (i.e. through a transparent USB-TCP proxy), a PCAP-compatible [3] timestamped packet stream (which can be used by Wireshark or other packet monitoring tools), or an ASCII stream in hexadecimal form. All these formats allow both remote monitoring and remote data logging which supplements local storage facilities.

Running the BaseStation software in promiscuous mode gives a better view on what is happening in the network. For the BaseStation-only way of operation unicast transmissions are usually used – packets addressed to the BaseStation node can be found in the incoming sniffer stream and for the logging purposes all the other packets can be filtered out. However, these extra packets can provide information about how messages are routed in the network or a very detailed data about with what transmission power must a message be sent in order to reach other nodes (the BaseStation or the next hop towards the BaseStation). For even better insight into the network operation, more than one sniffing station can be installed. As all the received packets are timestamped and marked with a tag unique to the sniffing station, all these packets can be later analyzed together. The NetServ program can connect simultaneously to USB and TCP sources of packets (i.e. the local sniffing node and the remote NetServ available through TCP) so all these sniffing points can be made available in just one place in real time.

Additionally, the BaseStation software provides a reverse channel through which clients can send WSN messages to the BaseStation. These messages can instruct the TelosBaseStation software to change its communication parameters (e.g. change the communication channel or transmit power) or can be sent to the WSN network. In this function the BaseStation software again acts as a proxy, allowing sending any packet, without checking or analyzing its contents – the only modification done is calculating the correct CRC at the end of the packet (which simplifies testing when packets are created manually from client connections by typing them in hex form or using copy and paste functions from some predefined examples). The typical usage however includes connecting to NetServ from a script written in Perl or Python and sending messages, then observing network response in the same script. This allows implementing automated test suites based on “send this packet”, “ex-

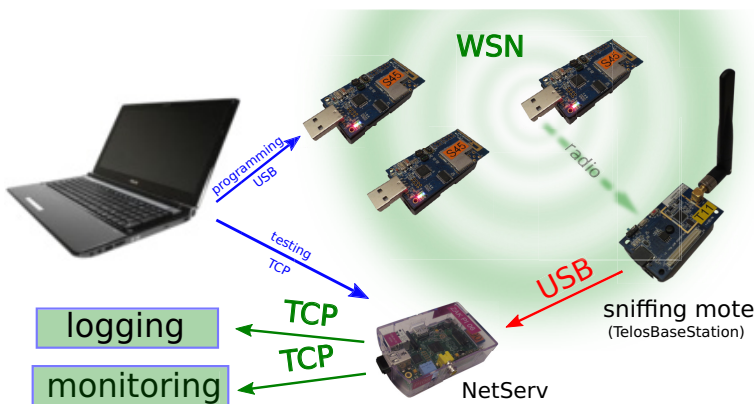


Fig. 1 Testing setup

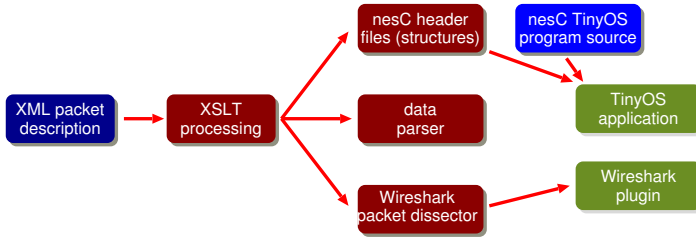


Fig. 2 Workflow of a WSN protocol development using formalized description

pect such response packet(s)” scenarios. In a similar manner a packet replay capability can be implemented. An external script can log a realistic communication scenario that is sniffed with a TelosBaseStation program and then recreate this communication many times to provide realistic and repeatable communication stress workload.

3 Formalized Methods of Protocol Description

Various tools can be used for observing the WSN traffic, starting with showing a raw packet stream, parsing this stream with custom-made interpretation scripts, or using specialized packet-inspection tools, such as tshark or Wireshark.

In order to facilitate fast development of communication and control protocols we have adopted a formalized way of specifying the data sent over the WSN. The format of all the packets is specified in form of an XML file, describing the fields and variants of messages to be sent. From this “master” file various resulting formats may be generated automatically (Fig. 2), including header files for nesC TinyOS programs, packet interpretation scripts, and Wireshark dissector code. This simplifies investigating the data captured from the air, allows for throughout analysis of how the network behaves, and also, prevents many bugs that stem from software incompatibilities.

Based on the XML packet description it is also possible to generate test cases for network testing or simplify the needed tools. So far, we have used mostly Python scripts for interacting with Netserv program in order to send messages to the WSN network and observe the network response. Examples of such tests are shown in the next section. The WSN traffic is made available by NetServ in different forms – simple byte-stream, PCAP-timestamped binary data or ASCII data in hexadecimal form. Testing scripts may use these data streams to allow automated verification of the tests (compatibility with our XML-based data description as one of the criteria). The PCAP-formatted stream may be connected to a Wireshark network analyzer allowing extended analysis of captured or live data.

4 Testcases

For implementing the BaseStation and serving the captured data with Net-Serv we have used Raspberry Pi embedded systems. Even though a typical laptop or a desktop computer with Linux operating system could be used for that purpose, a self-contained small system like BeagleBoard or Raspberry Pi is much better suited in a real WSN deployment, where it acts as a typical packet sink. As it does not need any keyboard or a display for functioning, it can be just plugged in to the mains supply and work, collecting the packets. When equipped with a WiFi dongle, it can immediately serve the collected data online. This is also useful in network testing scenarios, where various observation points can be created by placing several Raspberry Pi computers at various locations to observe the WSN radio traffic (Fig. 1).

Using this architecture we have performed numerous tests for our protocols. The most important ones were the tests of network self-organization, in which the network was switched back and forth between non-organized broadcast-mode operation and the organized mode with routing enabled. Two kinds of control messages were used for this purpose:

- **RESET** – Upon receiving a RESET message, a node sets an internal timer for a time period specified in the message payload. After this delay the node reboots. The delay is added to prevent multiple reboots if the node received the same RESET message several times, as it was retransmitted over the whole network. To test the proper functioning of this routine, we created a scenario where RESET messages were injected into the network with delay parameters varying from 1 ms to 60s. A misbehaviour of the code would manifest itself in the form of double resets, that is – two or

```
repeat {
  send HELO message
  collect traffic for n seconds
  send HELO message again
  collect more traffic
  if (broadcast packets found), abort
  if (no. of hops > threshold), abort // routing loops found
  if (bi-birectional paths), abort // routing loops found

  send RESET message
  collect traffic for n seconds
  if (unicast packets found), abort
  if (duplicate transmissions found), report multiple
    resets and abort
}
```

Fig. 3 Testing algorithm for HELO/RESET scenario

more messages with the same source address and sequence numbers would be seen if the reset was performed multiple times. Also, no unicast packets should be observed after the reset.

- HELO – a packet of type HELO is used to let other nodes know about their neighbours. This is necessary to construct a routing tree in each node. Each node is supposed to retransmit every HELO packet it receives, but only once (sequence numbers are used to detect duplicates) and after increasing the number of hops parameter. After exchanging a number of HELO packets, a routing tree should be created. Therefore, the test consists of sending a HELO packet from the BS and analysing communication after a set period of time. All the messages should from thereon *i*) be sent as unicasts, *ii*) be targeted at the BS and *iii*) there should be no routing loops.

The automated tests have been performed using algorithm specified in Fig. 3. Sample packets received in these tests are shown in Fig. 4. This example shows network behaviour for two consecutive RESET-HELO-HELO tests. Line 1 shows the RESET packet sent from the BaseStation running on node 47. Therefore the following packets from nodes 7 and 33 (lines 3-5) were sent as broadcasts. That is why the packet with seq_no 1 was received twice – directly from the originating node and retransmitted. However, the first packet from node 34 contained sequence number equal 3 (line 7), which means that packets with sequence numbers 1 and 2 were lost. After the HELO packet from the Base Station (line 11) the network started setting the routing tree. Lines 13-15 show packets which should be routed to the Base Station. Unfortunately there is a routing loop between nodes 7 and 33. If not for sniffing, such messages would never reach the Base Station. After another HELO packet in line 17 both nodes 7 and 33 correct their routes to the Base Station.

When the RESET-HELO-HELO test scenario is repeated (lines 23-39) nodes 7 and 33 properly establish the routing table, but node 34 apparently did not hear the first HELO packet and still sends broadcasts. After another HELO all nodes send messages directly to the Base Station.

Although the above description shows the detailed “manual” analysis of network behaviour, the automatic assessment can also detect protocol violations, as described in the testing algorithm (routing loops, wrong type of packets at a certain stage of the protocol, assertions on particular field values, etc.). Such automated tests can be run several times to test whether the communication protocols are resistant to errors such as lost or duplicate packets, packet collisions or unfortunate timings.

5 Future Work

From the testing perspective, our toolchain has functionality to:

- setup a testing network,
- capture WSN messages on a PC and analyze them,

- inject prepared messages into the WSN network through the gateway,
- perform automated tests by executing appropriate test scenarios.

Unfortunately, it may still be not enough to accurately observe the whole network, because the wireless links are unreliable. If we do not receive the expected packet, we can not be sure why it happened. Software and protocol errors or a high level of radio noise may be the reasons. Therefore it is necessary to develop methods which can log all activities on each node and then analyze the behaviour of the whole network after the test.

We intend to develop two solutions for this problem:

1. prepare a meta-protocol which can be used for sending debugging information, i.e. routing tables, packet reception rates, RSSI values, etc.
2. gather all debug information in node's Flash memory and then, after the experiment, collect the data via serial or radio connection.

Both methods have some disadvantages. The first one may interfere with the normal network operation and the protocols being tested, as the same

```

1 15:28:12.668 SEND RESET from=47, to=*, seq=11, delay=768ms, (7e 44 55 00 ff ff 00 00 0e 00 0c 01 05
   ff ff 00 2f 00 0b 00 00 00 00 03 00 39 2d 7e)
2 # 2 messages from node 33, one of them through node 7
3 15:29:02.702 from=33, to=*, via=7, seq=1, hops=2, photo=8293.15, path=[7,33]
4 02.720 from=33, to=*, via=33, seq=1, hops=1, photo=8293.15, path=[33]
5 02.813 from=7, to=*, via=7, seq=1, hops=1, photo=8426.67, path=[7]
6 # first packet coming from node 34
7 06.597 from=34, to=*, via=7, seq=3, hops=2, path=[7,34], photo=255.58
8 # some strange paths
9 16.716 from=34, to=*, via=7, seq=8, hops=3, path=[7,33,34], photo=137.33
10 16.736 from=7, to=*, via=33, seq=8, hops=2, path=[33,7], photo=8934.02
11 17.008 SEND HELLO from=47, to=*, seq=12, (bin: 7e 44 55 00 ff ff 00 00 0e 00 0c 01 05 ff ff 00 2f 00
   0c 00 00 00 00 03 00 7d 5d 34 7e)
12 # unicast messages, but routing tree is not ok.
13 18.398 from=34, to=47, via=33, seq=9, hops=2, path=[33,34], photo=244.14
14 18.416 from=33, to=47, via=7, seq=9, hops=3, path=[7,34,33], photo=8319.85
15 18.474 from=7, to=47, via=33, seq=9, hops=2, path=[33,7], photo=8888.24
16 # another HELLO
17 12.020 SEND HELLO from=47, to=*, seq=13, (7e 44 55 00 ff ff 00 00 0e 00 0c 01 05 ff ff 00 2f 00 0d 00
   00 00 00 03 00 1c 8c 7e)
18 12.955 from=33, to=47, via=33, seq=37, hops=1, path=[33], photo=8316.04
19 13.001 from=7, to=47, via=7, seq=37, hops=1, path=[7], photo=8937.84
20 14.920 from=33, to=47, via=33, seq=38, hops=1, path=[33], photo=8319.85
21 15.077 from=7, to=47, via=7, seq=38, hops=1, path=[7], photo=8899.69
22 # and 34 is somehow gone? No sign of it until RESET
23 43.439 SEND RESET from=47, to=*, seq=14, delay=1000ms, (7e 44 55 00 ff ff 00 00 0e 00 0c 01 08 ff ff
   00 2f 00 0e 00 00 00 00 03 e8 2c 63 7e)
24 # broadcast
25 45.257 from=7, to=*, via=33, seq=1, hops=2, photo=8361.82, path=[33,7]
26 45.284 from=33, to=*, via=33, seq=1, hops=1, photo=7667.54, path=[33]
27 49.172 from=34, to=*, via=7, seq=3, hops=2, path=[7,34], photo=251.77
28 51.596 SEND HELLO from=47, to=*, seq=15, (7e 44 55 00 ff ff 00 00 0e 00 0c 01 05 ff ff 00 2f 00 0f 00
   00 00 00 03 00 ff ec 7e)
29 # 34 did not get the HELLO message
30 53.007 from=34, to=*, via=33, seq=5, hops=2, path=[33,34], photo=205.99
31 54.955 from=33, to=47, via=33, seq=6, hops=1, path=[33], photo=8251.19
32 54.977 from=7, to=47, via=33, seq=6, hops=2, path=[33,7], photo=8857.73
33 ...
34 # another HELLO
35 15:30:32.947 SEND HELLO from=47, to=*, seq=16, (7e 44 55 00 ff ff 00 00 0e 00 0c 01 05 ff ff 00 2f 00
   10 00 00 00 00 03 00 6d 51 7e)
36 # and each node sends directly to BS (47)
37 34.065 from=33, to=47, via=33, seq=26, hops=1, path=[33], photo=8285.52
38 34.083 from=7, to=47, via=7, seq=26, hops=1, path=[7], photo=8831.02
39 34.097 from=34, to=47, via=34, seq=26, hops=1, path=[34], photo=469.21

```

Fig. 4 Sample packets observed by the testing script

underlying communications stack must be used for sending and receiving radio messages. The debugging messages may also be lost, and implementing a special reliable protocol with no impact on other messages sent through the network seems hard, if possible at all. The second method can be implemented only on a Flash-equipped WSN nodes and requires direct access to each node after the test, if serial/USB transmission is to be used. However, the log transfers are intended to happen after the testing has ended, so the radio transmission can also be used with reliable protocols, using acknowledgements and retransmissions while asking nodes one by one to dump their debugging logs over the radio. On the other hand, some nodes may still need manual intervention if during the test they have exhausted their power supplies.

6 Conclusions

When developing communication protocols for WSNs it is hard to effectively test the proper operation of the whole network. The system that we have developed extends the functions of a typical BaseStation node by allowing us to send arbitrary messages from the BS and observing the network response through capturing and analyzing the packets sent between WSN nodes. Test scenarios can be programmed in Python (or any other scripting/programming language) and use TCP client-server model for interaction with the network. Automated assessment of repeated test results is possible and the packet trail of network operation can be used for detailed post-factum analysis of the behaviour of network nodes. To make this trail complete, we intend to extend our system to provide local logging of node operation in node's Flash memories. After the experiment has ended, this data can be downloaded through a reliable communications link. The BaseStation-Netserv architecture of our system allows both logging such trails and providing a real-time access to WSN messages through TCP client-server connections. Wireshark packet analyzer can be used as a powerful tool for online or post-factum analysis of captured packets.

Acknowledgement. This work was supported by National Science Centre grant no. N 516 483740.

References

- [1] Berezowski, K.: The landscape of wireless sensing in greenhouse monitoring and control. *International Journal of Wireless & Mobile Networks (IJWMN)* 4(4), 141–154 (2012)
- [2] Hempstead, M., Welsh, M., Brooks, D.: TinyBench: the case for a standardized benchmark suite for TinyOS based wireless sensor network devices. In: 29th Annual IEEE International Conference on Local Computer Networks, pp. 585–586 (2004), doi:10.1109/LCN.2004.129

- [3] Jacobson, V., Lere, C., McCanne, S.: libpcap: Packet capture library. Lawrence Berkeley Laboratory, Berkeley (2009)
- [4] Langendoen, K.: Apples, Oranges, and Testbeds. In: 2006 IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), pp. 387–396 (2006), doi:10.1109/MOBHOC.2006.278578
- [5] Langendoen, K., Baggio, A., Visser, O.: Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In: 20th International on Parallel and Distributed Processing Symposium, IPDPS 2006 (2006), doi:10.1109/IPDPS.2006.1639412
- [6] Levis, P., Madden, S., Polastre, J., Szewczyk, R., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., Culler, D.: TinyOS: An operating system for sensor networks. In: Ambient Intelligence. Springer (2004)
- [7] Nazhandali, L., Minuth, M., Austin, T.: SenseBench: toward an accurate evaluation of sensor network processors. In: Proceedings of the IEEE International Workload Characterization Symposium, pp. 197–203 (2005), doi:10.1109/IISWC.2005.1526017
- [8] Słabicki, M., Wojciechowski, B., Surmacz, T.: Realistic model of radio communication in wireless sensor networks. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2012. CCIS, vol. 291, pp. 334–343. Springer, Heidelberg (2012)

Towards Precise Architectural Decision Models

Marcin Szlenk

Warsaw University of Technology,
Institute of Control and Computation Engineering,
Nowowiejska 15/19,00-665 Warsaw, Poland
m.szlenk@elka.pw.edu.pl

Abstract. One of the modern approaches for documenting software architecture is to show the architectural design decisions that led an architect to the final form of software architecture. However, decisions that have been made in such a process may need to be changed during further evolution and maintenance of the software architecture. The main reasons for these changes are new or changed requirements. In our team we have developed a graphical modelling notation for documenting architectural decisions, called Maps of Architectural Decisions, that can support the process of making changes in the software architecture. In this work we define a formal background for the controlled process of making changes in architectural decision models that are documented using that notation.

1 Introduction

Software architecture is developed as a result of numerous interrelated decisions. The architecture itself, these decisions and the context of these decisions create the architectural knowledge. Documenting architectural decisions is a new wave in architecture modelling [2, 12]. It deals with the representation, capture, management, and documentation of the design decisions made during architecting [9]. In the process of software maintenance and evolution, architectural decisions may undergo changes in response to new or changed requirements. Such decisions are often related with each other and changing one of them may affect the more extensive part of the decision model. Performing the changes in the architectural decision models in a rigorous way is an important problem we want to address in this work. The proposed solution is based on the modelling notation called Maps of Architectural Decisions (MAD) [13]. This work extends our previous work [11] mainly by introducing the concept of decision consistency (Sect. 5) and defining the formal metamodel of the MAD notation (Sect. 6).

2 Related Work and Motivation

A typical representation of architectural decisions are text records [1, 4, 12], that are sometimes accompanied with illustrating diagrams [3]. Many diagrammatic

(based on graphs) models have been also proposed as a way to represent architectural decision and decision making process in a more comprehensive way (see [10, 13, 14]). Graphical models and tools supporting architectural decisions and decision-making have been presented in [6, 10, 13].

Decision classifications have been developed to help to organise large sets of architectural decisions. Most influential classifications by Kruchten [8] (*existence, non-existence, property* and *executive* decisions) and Zimmermann [14] (*executive, conceptual, technology* and *vendor asset* decisions) substantially help to navigate through a set of architectural decisions. In both references, not only the categories of architectural decisions have been defined but also the possible kinds of relations between such decisions have been determined. In [8], Kruchten indicates as much as ten different kinds of relations between architectural decisions.

Another architectural decision model and diagrammatic notation (MAD) have been developed by our team and presented in [13]. MAD has been created to support architect-practitioners working on systems evolution. It does not impose any predefined classification or hierarchy of architectural decisions and assumes a limited number of relation kinds between architectural decisions. This makes the model of the decision process intuitive and easy to comprehend. To explain the choices made and capture their rationale, the entire decision situation is presented, including: the decision topic (or problem), considered design options, relevant requirements, the advantages and disadvantages of every considered option.

Although there are many approaches for representing and capturing architectural decisions, it seems that in all cases the following scenario is assumed: one has an initial set of requirements and according to these requirements the architectural decisions are made and documented. However, an important question arises: what activities should be done when the initial set of requirements changes but some or all of the decisions have been already captured in the model? Such a situation is quite usual during a project with iterative development [7] and typical for the maintenance phase when requirements for the next release of a system appear.

The new or changed requirements will usually lead to a changed decision model, but the main problem here is how to perform these changes in a controlled way and verify whether each change is justified by the context. The precise formal definitions would be welcomed here, as they open the further possibility of automatic verification. It seems that, so far, this problem has not been given much attention in the related works on architectural decision modelling. Thus, we would like to address it here in terms of models expressed in the MAD notation.

3 Maps of Architectural Decisions

MAD notation works similarly to mind maps used to present a problem structure graphically. The MAD models are built up of the following elements:

- *Decision problem* – represents the architectural issue being considered;
- *Connector* – in its basic form shows that one solved problem led an architect to the one indicated by an arrow (a "leads to" relation);

- *Solution* – represents a single solution to the architectural problem considered;
- *Requirement* – represents a requirement relevant to a given architectural problem;
- *Decision-maker* – represents a person or a group of people responsible for the resolution of a related architectural problem;
- *Pro or Con* – represents a single advantage or disadvantage of a given solution.

These elements have additional attributes, e.g. name, description, state, creation date and resolution date for the decision problem [13]. The most important elements of the notation are shown in Fig. 1.

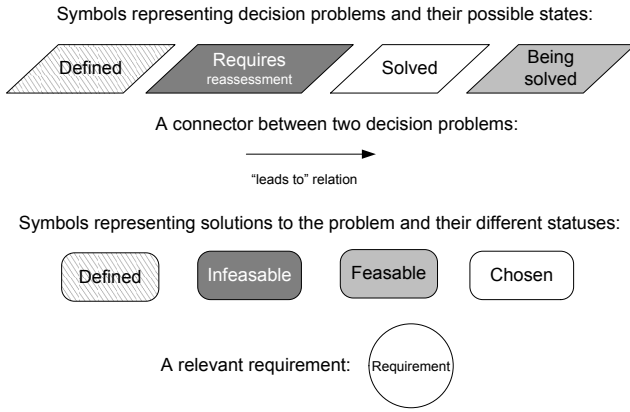


Fig. 1 The MAD notation

3.1 Decision Problem Life Cycle

The main objective of MAD is to show a decision making process and its progress. It introduces the concept of a decision problem's state, proposing four possibilities: *defined*, *being solved*, *solved*, and *requires reassessment* (see Fig. 1). The state transition rules have been defined for MAD using the concept of a decision problem's context. The context, in which the architectural decision problem is being considered, contains both the requirements and the decisions that have been already made [2]. To be more precise, not all the requirements and decisions made before should be considered here but, naturally, only these which are relevant to the given decision problem. In the MAD model the requirements are directly attached to decision problems they are relevant to, and the earlier decisions (chosen solutions), that led to the given problem, can be easily discovered by tracing the "leads to" relationship. In [11] the three definitions have been introduced:

Definition 1. (*Simplified MAD model*)

By a simplified MAD model we understand a tuple (*Problems*, *leadsTo*, *requirements*, *solution*), where:

- *Problems* is a set of decision problems,
- $leadsTo \subseteq Problems \times Problems$ is a set of pairs of decision problems connected through the "leads to" relation,
- $requirements(p)$ is a set of requirements relevant to the problem p , and
- $solution(p)$ is a single-element set containing a finally selected solution to the problem p (if the solution is not selected yet, then $solution(p)$ is undefined).

Definition 2. (*Reachability relation*)

Let M be a simplified MAD model and a relation $\succ \subseteq Problems \times Problems$ be the transitive closure of the relation $leadsTo$. The relation \succ will be called a reachability relation for the model M . If $p \succ q$ then we will say that the problem q is reachable from the problem p .

Definition 3. (*Context*)

Let M be a simplified MAD model, \succ be the reachability relation for the model M . The context of a problem $p \in Problems$ in M is defined as:

$$context(p) = requirements(p) \cup \bigcup_{q \succ p} solution(q).$$

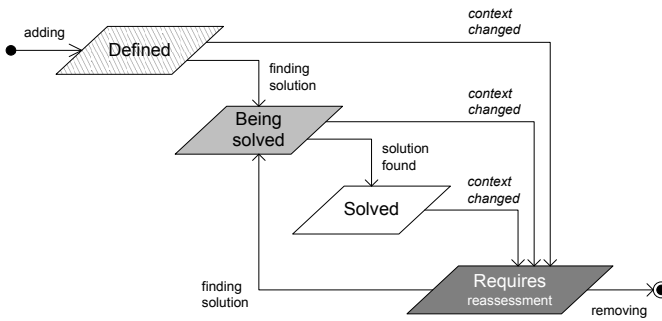


Fig. 2 The decision problem life cycle

Let us now discuss the life cycle of a decision problem in the MAD model. When a new decision problem is added to the model it is in the "defined" state. Next, when the possible solutions are being considered the problem is "being solved", and once one of the solutions connected to the problem becomes "chosen", the problem becomes "solved" and can lead to new problems. Whenever the problem's context is changing, the problem should automatically change its state into "requires reassessment". When the context of the decision problem has changed due to the changes of the previous decisions, the problem itself may not occur any more and in such a situation it should be removed from the model. The described decision problem life cycle is summarized in Fig. 2.

3.2 Model Rebuilding

The decision problem life cycle leads us to the rigorous process of making changes in MAD models in response to new requirements. When the new requirement appears, the software architect can decide whether it is relevant to one or more of the decisions captured in the MAD model. In the MAD model this new requirement will be then connected to proper decisions, changing at the same time their contexts and their status into “requires reassessment” as a result. The similar situation will occur when one of the requirements already existing in the model is changing.

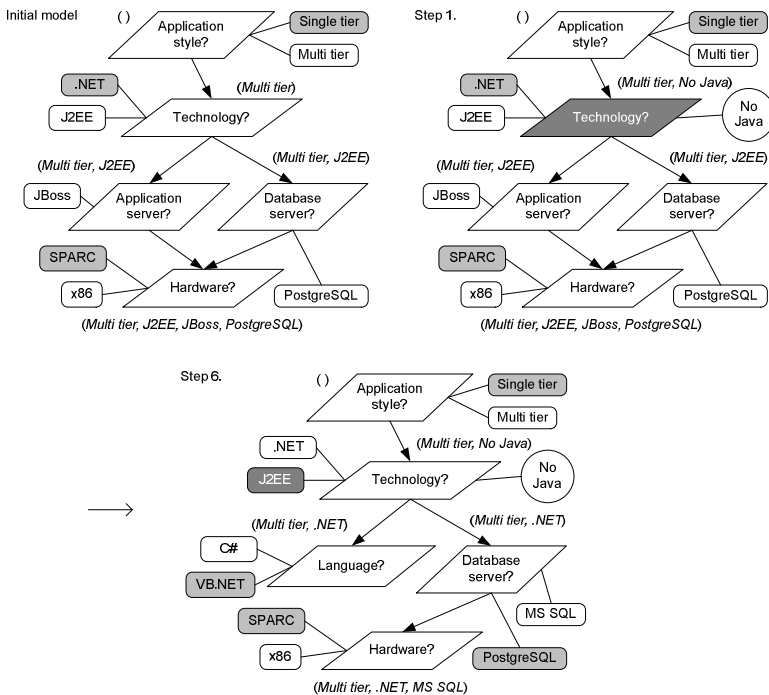


Fig. 3 Model rebuilding

Changing the status of any problem into “requires reassessment” may initiate the process of model rebuilding. An example of such a process has been presented in [11]. That process encompasses six steps. In Fig. 3, we present only the first step and the final result of the whole rebuilding process. The intermediate steps have been omitted. In the first step the new requirement telling that our company is moving from Java technology has been connected with the “Technology?” decision problem. For detailed description of the successive steps, please refer to [11].

4 Decision Consistency in MAD Models

As it has been shown in the previous section, the appearance of new requirements may result in a changed architectural decision model. Thus, the evolution process of a system may result in a sequence of architectural decision models, where two consecutive models reflect the single step of system evolution. Let us consider such a sequence of two architectural models (i.e. one evolution step) and call them M_1 and M_2 . Let p be one of the solved decision problems (a problem in the "solved" state) in the model M_1 . There are two possibilities:

- the problem p does not occur any more and is not present in M_2 ,
- the problem p does still occur and is present in both the models: M_1 and M_2 .

In the second case, if the problem p is also solved in the model M_2 then:

- the problem p has the same finally selected solution (a solution in the "chosen" state) in both of the models, or
- the problem p has a different finally selected solution, but in that case the contexts of p in M_1 and M_2 must be also different, otherwise there would not be any justification for changing a solution to the problem p (i.e. in the meantime p must have been in the "requires reassessment" state).

Let us now define the above relation between two MAD models formally.

Definition 4. (*Decision consistency relation*)

Let M_1 and M_2 be two simplified MAD models. We say that the decisions in M_1 and M_2 are *consistent* if they have the same finally selected solution or they contexts are different:

$$\forall p \in Problems_1 \cap Problems_2 \cdot \\ solution_1(p) \text{ is defined} \wedge solution_2(p) \text{ is defined} \Rightarrow \\ solution_1(p) = solution_2(p) \vee context_1(p) \neq context_2(p).$$

If M_2 is a result of rebuilding M_1 in a process of architecture evolution, the decisions in the model M_1 must be consistent with the decisions in the model M_2 in terms of the above definition. From the definition of the decision consistency relation it can be easily seen that this relation is both symmetric and reflexive.

5 Metamodel of MAD Notation

Although MAD notation offers constructions useful not only for documenting architectural decisions but also for making rigorous changes in a set of interrelating decisions, its biggest drawback when it comes to automatic verification of changes is its lack of precise metamodel. In this section, the metamodel of the part of MAD is presented. This metamodel is expressed in Alloy specification language [5], what allows for the analysis of the proposed metamodel using Alloy Analyzer tool.

Alloy is a declarative language for expressing structural constraints based on the first-order logic. An Alloy model is composed of sets (called ‘signatures’), relations in these sets (represented as signatures’ fields) and constraints over the sets and relations (called ‘facts’). For detailed description, please refer to [5] or to the Alloy project Web site <http://alloy.mit.edu>.

The partial MAD metamodel is presented in Fig. 4 and Fig. 5. The main elements of the MAD models (represented as the MAD signature) are: decision problems (the `DecisionProblem` signature), requirements (the `Requirement` signature) and solutions (the `Solution` signature). Every single problem and solution in the MAD model must be in one of the possible states represented here as the singleton subsets of the `ProblemState` and `SolutionState` signatures (compare with Fig. 1).

```

sig MAD {}
sig DecisionProblem {}
sig Requirement {}
sig Solution {}

abstract sig ProblemState {}
one sig DefinedProblem extends ProblemState {}
one sig RequiresReassessment extends ProblemState {}
one sig Solved extends ProblemState {}
one sig BeingSolved extends ProblemState {}

abstract sig SolutionState {}
one sig DefinedSolution extends SolutionState {}
one sig Infeasible extends SolutionState {}
one sig Feasible extends SolutionState {}
one sig Chosen extends SolutionState {}

```

Fig. 4 Signatures

Fig. 5 shows the different possible relations between models, problems, solutions, requirements and problems' and solutions' states, that take place in MAD models. For example, the `requirements` relation defines the set of requirements relevant to the problem (graphically connected to the problem). The following additional constraints have been added as the Alloy facts:

1. If the problem belongs to the model all of its preceding problems and successors also belong to that model;
2. The "leads to" relation cannot form a cycle;
3. Not more than one of considered solutions for the problem can be in the "chosen" state;

4. If the problem has a preceding problem, the preceding one must have been solved in the past (and then its solution generated the next problem) and can never be again in the "defined" state;
5. The solved problem must have a chosen solution.

```

sig MAD {
  problems: set DecisionProblem,
  problemState: problems -> one ProblemState,
  leadsTo: problems -> problems,
  requirements: problems -> Requirement,
  solutions: problems -> (Solution -> one SolutionState)
} {
  all p: problems | p.(leadsTo + ~leadsTo) in problems // 1.
  all p: problems | p not in p.^leadsTo // 2.
  all p: problems | lone s : Solution |
    s -> Chosen in solutions[p] // 3.
  all p: problems | all q: p.^leadsTo |
    problemState[q] != DefinedProblem // 4.
  all p: problems | problemState[p] = Solved =>
    one s: Solution | s -> Chosen in solutions[p] // 5.
}

```

Fig. 5 The MAD signature details

For the above MAD metamodel, the decision consistency relation, which was defined in Sect. 5, can be written in the form of an Alloy predicate as in Fig. 6.

```

fun context[m: MAD, p: DecisionProblem]: Requirement + Solution {
  m.requirements[p] + {s: Solution | some q: m.problems |
  p in q.(m.leadsTo) and s -> Chosen in m.solutions[q] }
}

fun solution[m: MAD, p: DecisionProblem]: Solution {
  {s: Solution | s -> Chosen in m.solutions[p]}
}

pred consistent [m1: MAD, m2: MAD] {
  all p: m1.problems & m2.problems |
    (m1.problemState[p] = Solved and
    m2.problemState[p] = Solved) =>
    (solution[m1,p] = solution[m2,p] or
    context[m1,p] != context[m2,p])
}

```

Fig. 6 Decision consistency relation

As it has been mentioned, Alloy specifications can be analyzed using the Alloy Analyzer tool. Such an analysis may be of two forms: *simulation*, which involves finding instances that satisfy a given property, or *checking*, which involves finding a counterexample—an instance that violates a given property. For the proposed metamodel, a number of instances have been generated with the Alloy Analyzer and these instances have been studied to assure both the correctness and completeness of the metamodel, i.e. that all the constraints have been captured and appropriately expressed.

6 Conclusion and Further Work

MAD notation has been originally developed as a simple tool for system architects to document architectural decisions [13]. It has appeared that this notation offers some constructions particularly useful for building the models in an iterative way, where new requirements appear after the whole or parts of the model were created [11]. In other words, the MAD notation may support the process of making changes in the software architecture during further evolution and maintenance of the software architecture.

MAD has been validated in the real life conditions of one of the largest telecom firms in Poland and a software tool supporting MAD has been also developed [13]. The tool has been designed as a diagram editor being an extension to MS Word. Unfortunately, the main problem we faced trying to extend this tool to support the process of rebuilding models was the lack of a precise MAD metamodel. In this work, such a metamodel has been proposed. After creating a tool based on this metamodel, further empirical evaluation of the concepts presented here will be conducted.

Acknowledgement. This work was sponsored by the Polish Ministry of Science and Higher Education under grant number 5321/B/T02/2010/39.

References

- [1] Ali Babar, M., Dingsøyr, T., Lago, P., van Vliet, H. (eds.): Software Architecture Knowledge Management: Theory and Practice. Springer (2009)
- [2] Bosch, J., Jansen, A.: Software Architecture as a Set of Architectural Design Decisions. In: Proc. 5th Working IEEE/IFIP Conference on Software Architecture (WICSA 2005), pp. 109–120. IEEE Computer Society (2005)
- [3] Capilla, R., Nava, F., Dueñas, J.C.: Modeling and Documenting the Evolution of Architectural Design Decisions. In: Proc. of the Second Workshop on Sharing and Reusing Architectural Knowledge Architecture, Rationale, and Design Intent, pp. 9–16. IEEE CS Press (2007)
- [4] Harrison, N.B., Avgeriou, P., Zdun, U.: Using Patterns to Capture Architectural Decisions. IEEE Software 24(4), 38–45 (2007)

- [5] Jackson, D.: *Software Abstractions: Logic, Language, and Analysis*, 2nd edn. MIT Press (2012)
- [6] Jansen, A., Avgeriou, P., van der Ven, J.S.: Enriching Software Architecture Documentation. *Journal of Systems and Software* 82(8), 1232–1248 (2009)
- [7] Kruchten, P.: *The Rational Unified Process - An Introduction*, 3rd edn. Addison-Wesley (2003)
- [8] Kruchten, P., Lago, P., van Vliet, H.: Building Up and Reasoning About Architectural Knowledge. In: Hofmeister, C., Crnković, I., Reussner, R. (eds.) *QoSA 2006*. LNCS, vol. 4214, pp. 43–58. Springer, Heidelberg (2006)
- [9] Kruchten, P., Capilla, R., Dueñas, J.C.: The Role of a Decision View in Software Architecture Practice. *IEEE Software* 26(2), 36–42 (2009)
- [10] Shahin, M., Liang, P., Khayyambashi, M.R.: Improving understandability of architecture design through visualization of architectural design decision. In: *Proceedings of the 2010 ICSE Workshop on Sharing and Reusing Architectural Knowledge*, pp. 88–95. ACM (2010)
- [11] Szlenk, M., Zalewski, A., Kijas, S.: Modelling architectural decisions under changing requirements. In: *Proc. Joint 10th Working Conference on Software Architecture & 6th European Conference on Software Architecture (WICSA/ECSA 2012)*, pp. 211–214. IEEE CS (2012)
- [12] Tyree, J., Akerman, A.: Architecture Decisions: Demystifying Architecture. *IEEE Software* 22(2), 19–27 (2005)
- [13] Zalewski, A., Kijas, S., Sokołowska, D.: Capturing Architecture Evolution with Maps of Architectural Decisions 2.0. In: Crnkovic, I., Gruhn, V., Book, M. (eds.) *ECSA 2011*. LNCS, vol. 6903, pp. 83–96. Springer, Heidelberg (2011)
- [14] Zimmermann, O., Koehler, J., Leymann, F., Polley, R., Schuster, N.: Managing architectural decision models with dependency relations, integrity constraints, and production rules. *Journal of Systems and Software* 82(8), 1249–1267 (2009)

Slot Selection Algorithms for Economic Scheduling in Distributed Computing with High QoS Rates

Victor Toporkov¹, Anna Toporkova²,
Alexey Tselishchev³, and Dmitry Yemelyanov¹

¹ National Research University “MPEI”,
ul. Krasnokazarmennaya 14, Moscow, 111250 Russia
ToporkovVV@mpei.ru, yemelyanov.dmitry@gmail.com

² National Research University Higher School of Economics,
Moscow Institute of Electronics and Mathematics,
Bolshoy Trekhsvyatitelsky per. 1-3/12, Moscow, 109028 Russia
atoporkova@hse.ru

³ CERN (European Organization for Nuclear Research),
CERN CH-1211 Genève 23 Switzerland
Alexey.Tselishchev@cern.ch

Abstract. In this work, we address the problem of slot selection and co-allocation for parallel jobs in distributed computing with non-dedicated resources. A single slot is a time span that can be assigned to a task, which is a part of a job. The job launch requires a co-allocation of a specified number of slots starting synchronously. The challenge is that slots associated with different CPU nodes of distributed computational environments may have arbitrary start and finish points that do not match. Some existing algorithms assigns a job to the first set of slots matching the resource request without any optimization (the first fit type), while other algorithms are based on an exhaustive search. In this paper, algorithms for effective slot selection of linear complexity are studied and compared with known decisions. The proposed algorithms allow overall increase in the quality of service (QoS) for each of the considered rates: job start time, finish time, runtime, CPU usage time and total cost of job execution.

1 Introduction

Economic mechanisms are used to solve problems like resource management and scheduling of jobs in a transparent and efficient way in distributed environments such as cloud computing and utility Grid [1-3]. A resource broker model [1-4] is decentralized, well-scalable and application-specific. It has two parties: node owners and brokers representing users. The simultaneous satisfaction of various application optimization criteria submitted by independent users is not possible due to several reasons and also can deteriorate such QoS rates as total execution

time of a batch of jobs or overall resource utilization. Another model is related to virtual organizations (VO) [5-7] with central schedulers providing job-flow level scheduling and optimization. VOs naturally restrict the scalability, but uniform rules for allocation and consumption of resources make it possible to improve the efficiency of resource usage and to find a trade-off between contradictory interests of different participants.

In [6, 7], we have proposed a hierarchical scheduling model which is functioning within a VO. The significant difference between the approach proposed in [6, 7] and well-known scheduling solutions for distributed environments such as Grids [2, 4, 5, 8, 9], e.g., gLite Workload Management System [8], is the fact that *the scheduling strategy* is formed on a basis of efficiency criteria. They allow reflecting economic principles of resource allocation by using relevant cost functions and solving a load balancing problem for heterogeneous resources. The metascheduler [6, 7] implements the economic policy of a VO based on local resource schedules. The schedules are defined as sets of slots coming from resource managers or schedulers in the resource domains. During each scheduling cycle the sets of available slots are updated and two problems have to be solved: 1) selecting an alternative set of slots (alternatives) that meet the requirements (resource, time, and cost); 2) choosing a slot combination that would be the efficient or optimal in terms of the whole job batch execution. To implement this scheduling scheme, first of all, one needs an algorithm for finding and co-allocating slot sets. An optimization technique for the second phase of this scheduling scheme was proposed in [6, 7].

First fit selection algorithms [10, 11] assign any job to the first set of slots matching the resource request conditions, while other algorithms use an exhaustive search, feature more than linear complexity, and may be inadequate for on-line use [12]. Moab scheduler [13] implements backfilling and during a slot window search does not take into account any additive constraints such as the minimum required storage volume or the maximum allowed total allocation cost. Moreover, backfilling does not support environments with non-dedicated resources. NWIRE system [4] performs a slot window allocation based on the user defined efficiency criterion under the maximum total execution cost constraint. However, the optimization occurs only on the stage of the best found offer selection.

In our previous works [14-16], two algorithms for slot selection AMP and ALP that feature linear complexity $O(m)$, where m is the number of available time-slots, were proposed. Both algorithms perform the search of the first fitting window without any optimization. AMP (Algorithm based on Maximal job Price), performing slot selection based on the maximum slot window cost, proved the advantage over ALP (Algorithm based on Local Price of slots) when applied to the above mentioned scheduling scheme. However, in order to accommodate an end user's job execution requirements, there is a need for more precise slot selection algorithms to consider various user demands along with the VO resource management policy.

In this paper, we propose algorithms for effective slot selection based on user defined criteria that feature linear complexity on the number of the available slots during the job batch scheduling cycle. The novelty of the proposed approaches consists of allocating a number of alternative sets of slots (alternatives). The proposed algorithms can be used for both homogeneous and heterogeneous resources.

The paper is organized as follows. Section 2 introduces a general scheme for searching alternative slot sets that are effective by the specified criteria. Then, in section 3, scheme implementations are proposed and considered. Section 4 contains simulation results for comparison of proposed and known algorithms. Section 5 summarizes the paper and describes further research topics.

2 General Scheme for Slot Selection Algorithms

In this section we consider a general scheme of an **A**lgorithm searching for **E**x-treme **P**erformance (AEP) and its implementation examples.

The launch of any job requires a co-allocation of a specified number of slots, as well as in the classic backfilling variation [13]. The task is to scan a list of m available slots and to select a window W of n parallel slots with a length of the required resource reservation time. The job resource requirements are arranged into a resource request containing a resource reservation time, characteristics of computational nodes (clock speed, RAM volume, disk space, operating system etc.) and the limitation on the selected window maximum cost. The total window cost is calculated as a sum of an individual usage cost of the selected slots. In addition, one can define a criterion on which the best matching window alternative is chosen. This can be a criterion crW for a minimum cost S , a minimum execution runtime or, for example, a minimum energy consumption. The algorithm parses a ranged list of all available slots subsequently for all the batch jobs. Higher priority jobs are processed first assuming that the job batch is presented as a list of jobs submitted for the scheduling cycle and ordered by their priority [6, 7].

Existing slot selection algorithms assign a job to the first set of slots matching the resource request conditions or use an exhaustive search. AEP is free of the obvious disadvantages of the exhaustive search and has linear complexity on the number of the slots available in the current scheduling cycle.

AEP can be compared to the algorithm of min/max value search in an array of flat values. The expanded window of size m “moves” through the ordered list of available slots. At each step any combination of n slots inside it (in case when $n \leq m$) can form a window that meets all the requirements to run the job. The effective on the specified criterion window of size n is selected from this m slots and compared with the results in the previous steps. By the end of the slot list the only solution with the best criterion crW value will be selected. The time length of an allocated window W is defined by the execution time of the task that is using the slowest CPU node. The algorithm proposed is processing a list of all available slots ordered by a non-decreasing start time during *the scheduling interval* denoting how far in the future the system may schedule resources. This condition

is required for a single sequential slot list scan and algorithm linear complexity on the number m of slots.

The scheme for an effective window search by the specified criteria can be represented as follows:

```

Input: slotList – a list of available slots
          job – a job for which the search is performed
Output: bestWindow – a window with the extreme criterion  $crW$  value
slotList = orderSystemSlotsByStartTime();
for each slot in slotList {
    if (!properHardwareAndSoftware(slot))
        continue;
    windowSlotList.add(slot);
    windowStartTime = slot.startTime;
    for each wSlot in windowSlotList {
        minLength = wSlot.Node.getWorkingTimeEstimate(job);
        if ((wSlot.endTime - windowStart) < minLength)
            windowSlotList.remove(wSlot);
    }
    if (windowSlotList.size >= job.nodesNeed) {
        curWindow = getBestWindow(windowSlotList);
        crW = getCriteriaValue(curWindow);
        if (crW > maxCriteriaValue) {
            maxCriteriaValue = crW;
            bestWindow = curWindow;
        }
    }
}

```

Finally, a variable `bestWindow` will contain an effective window by the given criterion.

3 AEP Implementation Examples

The need to choose alternative sets of slots for every batch job increases the complexity of the whole scheduling scheme [6, 7]. With a large number of available slots the search algorithm execution time may become inadequate.

Though it is possible to mention some typical optimization problems, based on the AEP scheme that can be solved with a relatively decreased complexity. These include problems of total job cost minimizing, total runtime minimizing, the window formation with the minimal start/finish time.

For the proposed AEP efficiency analysis the following algorithmic implementations were added to the simulation model [6, 7].

1. *AMP* – searching for slot windows with the earliest start time. This scheme was introduced in works [14-16]. The difference with the general AEP scheme is that the first suitable window will have the earliest possible start time.

2. *MinRunTime* – this algorithm performs a search for a single alternative with the minimum execution runtime. Given the nature of determining a window runtime, which is equal to the length of the longest composing slot, the following scheme may be proposed:

```

orderSlotsByCost(windowSlotList);
resultWindow = getSubList(0,n, windowSlotList);
extendWindow = getSubList(n+1,m, windowSlotList);
while(extendWindow.size > 0){
    longSlot = getLongestSlot(resultWindow);
    shortSlot = getCheapestSlot(extendWindow);
    extendWindow.remove(shortSlot);
    if((shortSlot.size < longSlot.size)&&
        (resultWindow.cost + shortSlot.cost < S)){
        resultWindow.remove(longSlot);
        resultWindow.add(shortSlot);
    }
}

```

As a result, the suitable window of the minimum time length with the total cost no more than S will be formed in a variable `resultWindow`. The algorithm described consists of the consecutive attempts to substitute the longest slot in the current window (the `resultWindow` variable) with another shorter one that will not be too expensive. In case when it is impossible to substitute the slots without violating the constraint on the maximum window allocation cost, the current `resultWindow` configuration is declared to have the minimum runtime.

3. *MinFinish* – searching for alternatives with the earliest finish time. This algorithm may be implemented using the runtime minimizing procedure presented above. Indeed, the expanded window has a start time t_{Start} equal to the start time of the last added suitable slot. The minimum finish time for a window on this set of slots is $(t_{Start} + minRuntime)$, where $minRuntime$ is the minimum window length. The value of $minRuntime$ can be calculated similar to the runtime minimizing procedure described above. Thus, by selecting a window with the earliest completion time at each step of the algorithm, the required window will be allocated in the end of the slot list.

4. *MinCost* – searching for a single alternative with the minimum total allocation cost on the scheduling interval. For this purpose in the AEP search scheme n slots with the minimum sum cost should be chosen. If at each step of the algorithm a window with the minimum sum cost is selected, at the end the window with the best value of the criterion crW will be guaranteed to have overall minimum total allocation cost at the given scheduling interval.

5. *MinProcTime* – this algorithm performs a search for a single alternative with the minimum total node execution time defined as a sum of the composing slots' time lengths. It is worth mentioning that this implementation is simplified and

does not guarantee an optimal result and only partially matches the AEP scheme, because a random window is selected.

6. *Common Stats, AMP* (further referred to as *CSA*) – the strategy for searching multiple alternatives using *AMP*. Similar to the general searching scheme [14-16], a set of suitable alternatives, disjointed by the slots, is allocated for each job. To compare the search results with the algorithms 1-5, presented above, only alternatives with the extreme value of the given criterion will be selected, so the optimization will take place at the selection process.

The criteria include *the start time, the finish time, the total execution cost, the minimum runtime and the processor time* used.

It is worth mentioning that all proposed AEP implementations have a linear complexity $O(m)$: algorithms “move” through the list of m available slots in the direction of non-decreasing start time without turning back or reviewing previous steps.

4 Simulation Studies of Slot Selection Algorithms

The goal of the experiment is to examine AEP implementations: to analyze alternatives search results with different efficiency criteria, to compare the results with *AMP* and to estimate the possibility of using in real systems considering the algorithm execution time. A simulation framework [6, 7] was configured in a special way in order to study and to analyze the algorithms presented. The core of the system includes several components that allow generating the initial state of the distributed environment on the given scheduling time interval, a batch with user jobs and implements the alternative slot search.

In each experiment a generation of the distributed environment that consists of 100 CPU nodes was performed. The performance rate for each node was generated as a random integer variable in the interval [2; 10] with a uniform distribution. The resource usage cost was formed proportionally to their performance with an element of normally distributed deviation in order to simulate a free market pricing model [1-4]. The level of the resource initial load with the local and high priority jobs at the scheduling interval [0; 600] was generated by the hypergeometric distribution in the range from 10% to 50% for each CPU node. Based on the generated environment the algorithms performed the search for a single initial job that required an allocation of 5 parallel slots for 150 units of time. The maximum total execution cost according to user requirements was set to 1500. This value generally will not allow using the most expensive (and usually the most efficient) CPU nodes. The relatively high number of the generated nodes has been chosen to allow *CSA* to find more slot alternatives. Therefore more effective alternatives could be selected for the searching results comparison based on the given criteria. The results of the 5000 simulated scheduling cycles are presented in Fig. 1. The obtained values of the *total job execution cost* are as follows: *AMP* – 1445,2; *minFinish* – 1464,2; *minCost* – 1027,3; *minRuntime* – 1464,9; *minProcTime* – 1342,1; *CSA* – 1352.

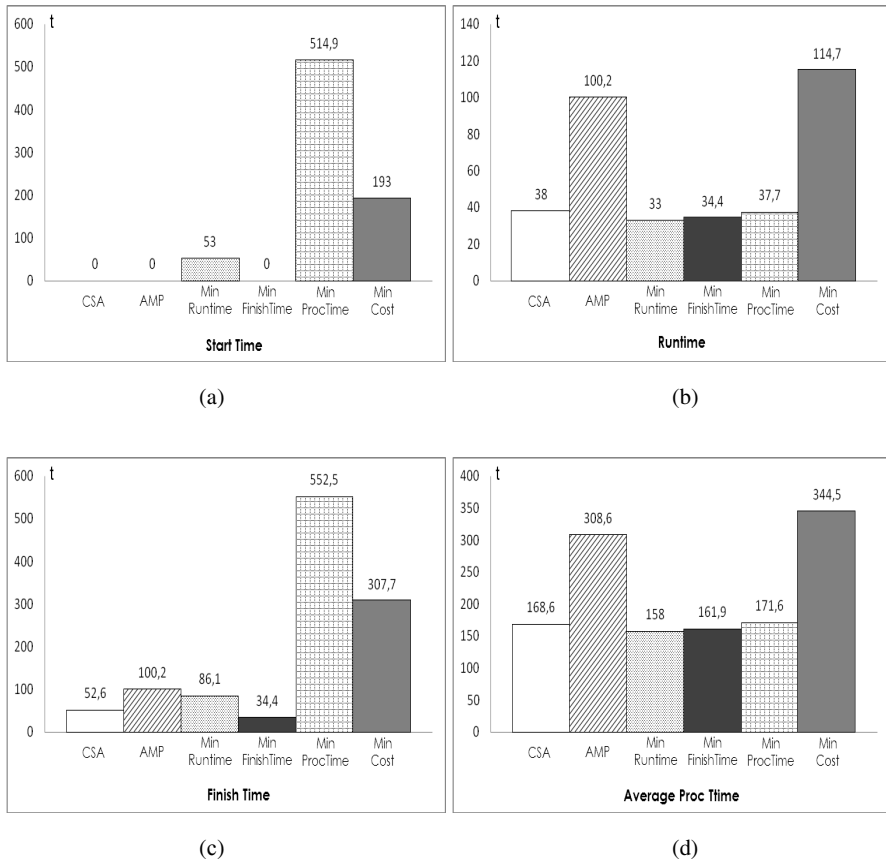


Fig. 1 Average start time (a), runtime (b), finish time (c), and CPU usage time (d)

Each full AEP implementation was able to obtain the best result in accordance with the given criterion: start time (Fig. 1 (a)); runtime (Fig. 1 (b)); finish time (Fig. 1 (c)); CPU usage time (Fig. 1 (d)). Besides, a single run of AEP had an advantage of 10%-50% over suitable alternatives found by AMP with a respect to the specified criterion. According to the simulation results, on one hand, the best scheme with top results in start time, finish time, runtime and CPU usage time was *minFinish*. Though in order to obtain such results the algorithm spent almost all user specified budget (1464 of 1500). On the other hand, the *minCost* algorithm was designed precisely to minimize execution expenses and provides 43% advantage over *minFinish* (1027 of 1500), but the drawback is a more than modest results by other criteria considered.

The important factor is a complexity and an actual working time of the proposed algorithms.

Table 1 shows the actual algorithm execution time in milliseconds measured depending on the number of CPU nodes. The simulation was performed on a regular PC workstation with Intel Core i3 (2 cores @ 2.93 GHz), 3GB RAM on JRE 1.6, and 1000 separate experiments were simulated for each value of the processor nodes numbers {50, 100, 200, 300, 400}. The CSA strategy has the longest working time that on the average almost reaches 3 seconds when 400 nodes are available. AEP implementations feature a quadratic complexity.

Table 1 Actual algorithms execution time (in ms) measured depending on the nodes number

CPU nodes number:	50	100	200	300	400
<i>CSA:Alternatives Num</i>	25.9	57	128.4	187.3	252
<i>CSA per Alt</i>	0.33	0.99	3.16	6.79	11.83
<i>CSA</i>	8.5	56.5	405.2	1271	2980.9
<i>AMP</i>	0.3	0.5	1.1	1.6	2.2
<i>MinRunTime</i>	3.2	12	45.5	97.2	169.2
<i>MinFinishTime</i>	3.2	12	45.1	96.9	169
<i>MinProcTime</i>	1.5	5.2	19.4	42.1	74.1
<i>MinCost</i>	1.7	6.3	23.6	52.3	91.5

Table 2 contains the algorithms working time in milliseconds measured depending on the scheduling interval length.

Table 2 Algorithms working time (in ms) measured depending on the scheduling interval length

Scheduling interval length:	600	1200	1800	2400	3000	3600
Number of slots:	472.6	779.4	1092	1405.1	1718.8	2030.6
<i>CSA:Alternatives Num</i>	57	125.4	196.2	269.8	339.7	412.5
<i>CSA per Alt</i>	0.95	1.91	2.88	3.88	4.87	5.88
<i>CSA</i>	54.2	239.8	565.7	1045.7	1650.5	2424.4
<i>AMP</i>	0.5	0.82	1.1	1.44	1.79	2.14
<i>MinRunTime</i>	11.7	26	40.9	55.5	69.4	84.6
<i>MinFinishTime</i>	11.6	25.7	40.6	55.3	69	84.1
<i>MinProcTime</i>	5	11.1	17.4	23.5	29.5	35.8
<i>MinCost</i>	6.1	13.4	20.9	28.5	35.7	43.5

Overall 1000 single experiments were conducted for each value of the interval length {600, 1200, 1800, 2400, 3000, 3600} and for each considered algorithm an average working time was obtained. The simulation parameters and assumptions were the same as described earlier in this section, apart from the scheduling interval length. A number of CPU nodes was set to 100. Similarly to the previous experiment, CSA had the longest working time (about 2.5 seconds with the

scheduling interval length equal to 3600 model time units), which is mainly caused by the relatively large number of the formed execution alternatives (on the average more than 400 alternatives on the 3600 interval length). Analyzing the presented values it is easy to see that all proposed algorithms have a linear complexity with the respect to the length of the scheduling interval and, hence, to the number of the available slots. The *minProcTime* strategy stands apart and represents a class of simplified AEP implementations with a noticeably reduced working time. And though the scheme compared to other considered algorithms, did not provide any remarkable results, it was on the average only 2% less effective than the *CSA* scheme by the dedicated CPU usage criterion (see Fig. 1 (d)). At the same time its reduced complexity and actual working time allow to use it in a large wide scale distributed systems when other optimization search algorithms prove to be too slow (see Tables 1 and 2).

5 Conclusions and Future Work

In this work, we address the problem of slot selection and co-allocation for parallel jobs in distributed computing with non-dedicated resources. For this purpose AMP and AEP approaches were proposed and considered. Specific AEP scheme implementations with a reduced (compared to a general scheme) complexity were introduced. Each of the algorithms possesses a linear complexity on a total available slots number and a quadratic complexity on a CPU nodes number. The advantage of AEP implementations over the general *CSA* scheme was shown for each of the considered QoS rates: start time, finish time, runtime, CPU usage time and total cost. In our further work we will refine resource co-allocation algorithms in order to integrate them with scalable co-scheduling strategies [6, 7].

Future research will be focused on further AEP research, its integration with the whole batch scheduling approach, and mainly on its influence on job-flows execution efficiency.

Acknowledgements. This work was partially supported by the Council on Grants of the President of the Russian Federation for State Support of Leading Scientific Schools (SS-316.2012.9), the Russian Foundation for Basic Research (grant no. 12-07-00042), and by the Federal Target Program “Research and scientific-pedagogical cadres of innovative Russia” (state contract no. 16.740.11.0516).

References

- [1] Lee, Y.C., Wang, C., Zomaya, A.Y., Zhou, B.B.: Profit-driven scheduling for cloud services with data access awareness. *J. of Parallel Distributed Computing* 72(4), 591–602 (2012)
- [2] Garg, S.K., Yeo, C.S., Anandasivam, A., Buyya, R.: Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. *J. of Parallel and Distributed Computing* 71(6), 732–749 (2011)

- [3] Garg, S.K., Buyya, R., Siegel, H.J.: Scheduling parallel applications on utility Grids: time and cost trade-off management. In: 32nd Australasian Computer Science Conference, pp. 151–159 (2009)
- [4] Ernemann, C., Hamscher, V., Yahyapour, R.: Economic scheduling in Grid computing. In: Feitelson, D.G., Rudolph, L., Schwiegelshohn, U. (eds.) JSSPP 2002. LNCS, vol. 2537, pp. 128–152. Springer, Heidelberg (2002)
- [5] Kurowski, K., Nabrzyski, J., Oleksiak, A., et al.: Multicriteria aspects of Grid resource management. In: Nabrzyski, J., Schopf, J.M., Weglarz, J. (eds.) Grid Resource Management. State of the Art and Future Trends, pp. 271–293. Kluwer Acad. Publ. (2003)
- [6] Toporkov, V., Tselishchev, A., Yemelyanov, D., Bobchenkov, A.: Composite scheduling strategies in distributed computing with non-dedicated resources. *Procedia Computer Science* 9, 176–185 (2012)
- [7] Toporkov, V., Tselishchev, A., Yemelyanov, D., Bobchenkov, A.: Dependable strategies for job-flows dispatching and scheduling in virtual organizations of distributed computing environments. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability*. AISC, vol. 170, pp. 289–304. Springer, Heidelberg (2012)
- [8] Cecchi, M., Capannini, F., Dorigo, A., et al.: The gLite workload management system. *Journal of Physics: Conference Series* 219(6), 062039 (2010)
- [9] Yu, J., Buyya, R., Ramamohanarao, K.: Workflow scheduling algorithms for Grid computing. In: Xhafa, F., Abraham, A. (eds.) *Metaheuristics for Scheduling in Distributed Computing Environments*. SCI, vol. 146, pp. 173–214. Springer, Heidelberg (2008)
- [10] Aida, K., Casanova, H.: Scheduling mixed-parallel applications with advance reservations. In: *Proc. of the 17th IEEE Intern. Symposium on High-Performance Distributed Computing*, pp. 65–74. IEEE CS Press, New York (2008)
- [11] Elmroth, E., Tordsson, J.: A standards-based Grid resource brokering service supporting advance reservations, coallocation and cross-Grid interoperability. *J. of Concurrency and Computation: Practice and Experience* 25(18), 2298–2335 (2009)
- [12] Takefusa, A., Nakada, H., Kudoh, T., Tanaka, Y.: An advance reservation-based co-allocation algorithm for distributed computers and network bandwidth on qoS-guaranteed grids. In: Frachtenberg, E., Schwiegelshohn, U. (eds.) JSSPP 2010. LNCS, vol. 6253, pp. 16–34. Springer, Heidelberg (2010)
- [13] Moab Adaptive Computing Suite,
<http://www.adaptivecomputing.com/products/moab-adaptive-computing-suite.php>
- [14] Toporkov, V., Toporkova, A., Bobchenkov, A., Yemelyanov, D.: Resource selection algorithms for economic scheduling in distributed systems. *Procedia Computer Science* 4, 2267–2276 (2011)
- [15] Toporkov, V., Yemelyanov, D., Toporkova, A., Bobchenkov, A.: Resource co-allocation algorithms for job batch scheduling in dependable distributed computing. In: Zamojski, W., Kacprzyk, J., Mazurkiewicz, J., Sugier, J., Walkowiak, T. (eds.) *Dependable Computer Systems*. AISC, vol. 97, pp. 243–256. Springer, Heidelberg (2011)
- [16] Toporkov, V., Bobchenkov, A., Toporkova, A., Tselishchev, A., Yemelyanov, D.: Slot Selection and Co-allocation for Economic Scheduling in Distributed Computing. In: Malyshev, V. (ed.) *PaCT 2011*. LNCS, vol. 6873, pp. 368–383. Springer, Heidelberg (2011)

K-Induction Based Verification of Real-Time Safety Critical Systems

Tamás Tóth, András Vörös, and István Majzik

DMIS, Budapest University of Technology and Economics,
Magyar Tudósok krt. 2., Budapest, Hungary
{toth.tamas,vorosa,majzik}@inf.mit.bme.hu

Abstract. Nowadays, safety critical systems are often complex, real-time systems requiring formal methods to prove the correctness of their behavior. This work presents a framework that supports modeling and model checking such systems. We adapted an existing formalism to provide better modeling and model checking support. Using this formalism, we extended a k-induction based model checking approach: we defined a procedure to handle both safety and liveness properties, and developed methods to find invariants. We implemented a toolchain for this workflow and evaluated our methods in an industrial case study.

1 Introduction

The formal proof of correctness of the behavior of safety critical systems is a challenging task as these systems are often fault-tolerant, real-time distributed systems with time-dependent data processing. We faced this problem in checking the correctness of an industrial protocol, the ProSigma SCAN protocol developed by Prolan that is responsible for safe transmission of the status of field modules to a control center. We addressed the verification problem by formal modeling and model checking. Our first attempts using several classic modeling formalisms and model checking tools (e.g. timed automata [1]) revealed difficulties. First, the use and processing of time-stamps (that was included in the protocol) was either not allowed, or resulted in an infinite state space that could not be handled. Accordingly, we turned towards formalisms that support induction based proofs (the so-called k-induction [10]). However, k-induction based techniques supported only the verification of safety properties (invariants). This way we decided to extend the capabilities of these techniques to support the checking of liveness properties. Second, the formalism that supported k-induction required quite low-level transition systems that were not easy to construct and understand by engineers. Accordingly, we decided to provide a higher-level formalism (so-called calendar systems) that is more easy to use, and can be automatically mapped to the underlying lower level formalism. This formalism proved to be advantageous to find modeling problems by static analysis, and identify invariants that are often required in k-induction based proofs.

In our work we introduce the framework that supports these achievements. After summarizing the formal background in Section 2, the new results are presented. The adapted formalism is introduced in Section 3. The extensions of the k-induction based model checking are discussed in Section 4. The tool support we provided is summarized in Section 5. Finally, we present the validation of our approach by verifying the industrial protocol that motivated our research.

2 Transition Systems and Checking of ω -Regular Properties

Transition Systems. One of the basic ways of describing discrete event systems is by the means of transition systems. A transition system is a tuple $TS = (S, \rightarrow, I, AP, L)$ where S is a set of states, $\rightarrow \subseteq S \times S$ is a transition relation over states, $I \subseteq S$ is the set of initial states, AP is the set of atomic propositions and $L: S \rightarrow AP$ is the labeling function that assigns propositions to states.

The behavior of a system is described by the means of traces. An (infinite) trace of the system is sequence of sets of atomic propositions $\pi = L(s_0)L(s_1)L(s_2)\dots$, where each $s_i \in S$ is a state of the system, $s_0 \in I$ is an initial state and for all s_i, s_{i+1} there is a transition $(s_i, s_{i+1}) \in \rightarrow$. The set of all infinite traces of the systems is denoted as $Traces(TS)$, which is an ω -language over the alphabet 2^{AP} .

Formalizing Requirements. To formalize requirements for a system, a possible approach is to specify the traces that reflect the fault-free behavior of the system. Such a requirement (a property) is – like the traces of the system – also an ω -language. In model checking, ω -regular languages (including properties formulated in linear temporal logic) are commonly used, that are equivalent to the languages accepted by nondeterministic Büchi-automata.

From a syntactic point of view, a nondeterministic Büchi-automaton is identical to a nondeterministic finite automaton, formally $A = (Q, \Sigma, \Delta, Q_0, F)$ where Q is a finite set of states, Σ is a finite set (alphabet), $\Delta: Q \times \Sigma \times Q$ is the transition relation, $Q_0 \subseteq Q$ is the set of initial states and $F \subseteq Q$ is the set of accepting states.

A *run* of the automaton over the word $\alpha = a_0 a_1 \dots$ (with each $a_i \in \Sigma$) is an infinite sequence of states $q_0 q_1 q_2 \dots$ where $q_0 \in Q_0$ is an initial state, each $q_i \in Q$ a state and for all q_i, q_{i+1} there is a transition $(q_i, a_i, q_{i+1}) \in \Delta$. Automaton A accepts those runs in which one of the accepting states occurs infinitely often, and accepts a word if there is an accepting run over it.

The language accepted by the Büchi-automaton is denoted as $L(A)$. The class of ω -regular languages is closed under complementation, thus for all automata A there exist an automaton \bar{A} for that $L(\bar{A}) = \overline{L(A)}$.

Model Checking of ω -Regular Properties. Given a transition system TS and a nondeterministic Büchi-automaton A , the model checking problem is to evaluate

whether $Traces(TS) \subseteq L(A)$, that is, to check that each trace of the system represents a fault-free behavior. Equivalently, one can check if $Traces(TS) \cap L(\bar{A}) = \emptyset$.

According to automata theoretic model checking introduced in [11], this can be performed as the following. From a transition system TS and nondeterministic Büchi-automaton A one can produce the transition system $TS' = TS \otimes A$ with the signature $TS' = (S \times Q, \rightarrow', I', AP', L')$ where $AP' = Q$, $L'(\langle s, q \rangle) = \{q\}$, $I' = \{\langle s_0, q \rangle : s_0 \in I, \exists q_0 \in Q_0 : (q_0, L(s_0), q) \in \Delta\}$ and for each transition $(s, t) \in \rightarrow$ of TS and $(p, L(t), q) \in \Delta$ of A the system has a transition $(\langle s, p \rangle, \langle t, q \rangle) \in \rightarrow'$.

Since the language $Traces(TS \otimes \bar{A})$ contains the runs of \bar{A} over the traces of TS , the model checking problem can be solved by checking that for all of its traces holds that eventually only nonaccepting states follow (which itself induces an ω -regular property, a so-called persistence property).

For finite state spaces this is a special circle detection problem since in that case each infinite run can be represented by a finite prefix forming a lasso [3]. A counterexample for the property is then a lasso-shaped run for which there is an accepting state (more precisely, a state labeled with an accepting state) inside the loop of the lasso, and the absence of such lassos is a proof for the property.

Model Checking Invariant Properties with k-Induction. *K-induction* [10] is the generalization of ordinary induction that enables model checking invariant properties of transition systems. Like standard (1-)induction, it enables the handling of infinite state spaces but is for most systems stronger and thus enables the proving of weaker invariants.

We use the following notations. Let TS be a transition system over a set of atomic propositions AP and P a propositional formula over the same set. A state $s \in S$ of TS satisfies the formula P (denoted as $P(s)$) if the label $L(s)$ of the state makes the formula evaluate to true. Using this notation, the k-induction principle for transition systems is the following (given a $k \in \mathbb{N}$).

- Base case: showing that along all paths $s_0 s_1 \dots s_{k-1}$, where $s_0 \in I$ is an initial state, $P(s_i)$ holds for all $0 \leq i < k$.
- Induction step: showing that along all paths $s_0 s_1 \dots s_{k-1}$ where $P(s_i)$ holds for all $0 \leq i < k$, then for every $s_k \in S$ such that $(s_{k-1}, s_k) \in \rightarrow$ the proposition $P(s_k)$ holds as well. If a counterexample is found, the proof with induction (in depth k) fails.

In many cases, for successful (or efficient) verification of the system, *supporting invariants* are required. Given an invariant L , it suffices to consider during the induction step only those paths where $L(s_i) \wedge P(s_i)$ holds along the path since only L -states are reachable. Thus the use of proper invariants allows avoiding counterexamples that appear in the induction step.

3 The Modeling Formalism

Inspired by the paper [6], we adapted for our purposes the formalism of *calendar automata*, since it supports the modeling of time-dependent behaviour, the use of time-stamps, and k-induction based model checking.

Calendar automata is a formalism for describing timed systems as transition systems. Its main idea is based on that of discrete event simulation: instead of clocks of timed automata (that store the time elapsed since a past event), it uses variables to store events scheduled to occur at a point of time in the future. Although this way time progresses to infinity, resulting in an infinite state space, the formalism is easy to handle with induction.

Time progress is modeled as follows. A calendar automaton has a set of timeouts that stores local events and an event calendar for events the automata schedule for each other. A discrete transition may update timeouts to future values or dispatch events to the calendar, again, scheduled to occur in the future. Such transitions must also consume a current event from the calendar or update a current timeout to prevent instantaneous loops. Time progress transitions are enabled if no current events are available, that is, if the time is lower than any point in time when an event is scheduled to occur. If so, they update time to the time value of the next event. Provided this behavior, the time value of events may never be lower than the current time, and maximal time progress is guaranteed.

To increase model checking performance, we applied two modifications.

- To shorten paths in the state space we adapted the method of merging discrete and time progress transitions introduced in [9]. By doing so, the induction depth needed to verify properties is significantly decreased.
- To eliminate the need for updating momentarily irrelevant timeouts to future values, we modified time progress semantics so that only a valid subset of timeouts is taken into account by determining time value for the next step. This is performed by enabling the possibility for transitions to explicitly validate and invalidate timeouts, thus marking the set of timeouts that are taken into account. This way a great deal of nondeterminism and deadlocks are eliminated, thus improving the performance of the verification.

Over the modified semantics we developed a higher level formalism, the *calendar system* formalism that makes modeling easier, yet is still suitable for describing a broad range of systems. The next paragraphs describe its syntax and semantics in detail.

Syntax of Calendar Systems. Let Δ denote the set of subintervals of $(0;+\infty)$ where the endpoints are natural numbers. A calendar system is a tuple

$$CS = (Loc, M, T, \rightarrow, Loc_0, T_0, T_{valid,0}, AP, L)$$

- where Loc is a finite set of control locations,
- M is a finite set of messages that contains the empty message m_0
- T is a finite set of timeouts

- $\rightarrow \subseteq Loc \times (M \cup T) \times (M \times \Delta) \times 2^{(T \times \Delta)} \times 2^T \times Loc$ is the transition relation
- $Loc_0 \subseteq Loc$ is the set of initial locations
- $T_0 : T \rightarrow \Delta$ is a function that assigns possible initial values to timeouts
- $T_{valid,0} \subseteq T$ is the initial set of valid timeouts
- AP is the set of atomic propositions
- $L : Loc \rightarrow AP$ is the labeling function

For an element $(\ell, e, \tilde{m}, \tilde{T}, T^-, \ell') \in \rightarrow$ of the transition relation, $e \in (M \cup T)$ is the triggering event of the transition (that is either the reception of a message or a timeout), $\tilde{m} \in (M \times \Delta)$ is a message sending action, $\tilde{T} \in 2^{(T \times \Delta)}$ is the set of timeout updating actions, and $T^- \in 2^T$ is the set of invalidated timeouts.

The well-formedness constraints are the following: $M \cap T = \emptyset$ and the empty message m_0 is never included in an event. Furthermore, for a timeout $x \in T$ a transition may update the timeout with an action $\tilde{x} = \langle x, I \rangle \in \tilde{T}$ or may invalidate that timeout ($x \in T^-$), but not both.

Semantics of Calendar Systems. Let Var be a set of variables and $Eval(Var)$ be the set of all possible evaluations over the variables. An evaluation $\sigma \in Eval(Var)$ is a function that assigns each variable $x \in Var$ a value from its respective domain $dom(x)$. The semantics of a calendar system CS is defined by the transitions system $TS(CS) = (Loc \times Eval(Var), \rightarrow', I', AP, L')$

- where $Var = \{t, x_1, x_2, \dots, T_{valid}, C\}$ with t representing the current time, each $x_i \in T$ storing a timeout value, T_{valid} storing the currently valid timeouts and C representing the event calendar. The domain of t, x_1, x_2, \dots is the set of non-negative real numbers. The domain of T_{valid} is 2^T , and the possible values of C are finite multisets of the form $c = \{\langle m_i, t_i \rangle : m_i \in M, t_i \geq 0\}$. The capacity of the calendar is bounded, that is, it can store only a limited number of events.
- $\rightarrow' \subseteq (Loc \times Eval(Var)) \times (Loc \times Eval(Var))$ is the transition relation.
- $I' \subseteq Loc \times Eval(Var)$ is the set of initial states. For each initial state $\langle \ell, \sigma \rangle \in I'$ the following constraints hold: $\ell \in Loc_0$, $\sigma(t) = 0$, $\sigma(x_i) \in T_0(x_i)$, $\sigma(T_{valid}) = T_{valid,0}$ and $\sigma(C) = \emptyset$.
- $L'(\langle \ell, \sigma \rangle) = L(\ell)$ is the labeling function

For the description of the semantics of the transition relation, we use the following notations (as in [6]). For a set of timeouts $T' \subseteq T$, let $\min(\sigma(T')) = \min\{\sigma(x) : x \in T'\}$ denote the minimum of timeouts in T' for a given evaluation ($+\infty$ for the empty set). Similarly, $\min(\sigma(C)) = \min\{t_i : \langle m_i, t_i \rangle \in C\}$

denotes the smallest time of occurrence between the calendar events (again, $+\infty$ for the empty calendar). Given a real u , we denote by $Ev_u(c) = \{\langle m_i, t_i \rangle : t_i = u, \langle m_i, t_i \rangle \in c\}$ the set of events in the calendar scheduled to occur at u .

Let $(\ell, e, \tilde{m}, \tilde{T}, T^-, \ell') \in \rightarrow$ be a transition of CS , with $\tilde{m} = \langle m, I \rangle$. Then for a transition $(\langle \ell, \sigma \rangle, \langle \ell', \sigma' \rangle) \in \rightarrow'$ of $TS(CS)$ the following rules apply:

- $\sigma'(t) = \min\{\min(\sigma'(T_{valid})), \min(\sigma'(C))\}$ - time advances to the next event, that is either determined by a valid timeout, or an event in the calendar.
- If $\langle x, J \rangle \in \tilde{T}$ then $\sigma'(x) = \sigma'(t) + d$ for some $d \in J$ - each timeout updating action sets a timeout to a future value. Other than that, the action implicitly validates the timeout if it was invalid before, that means $x \in \sigma'(T_{valid})$.
- If $x \in T^-$ then $x \notin \sigma'(T_{valid})$ - that means the timeout is invalidated by the transition. Timeouts that were neither validated nor invalidated remain as they were.
- If $e \in T$ - the transition is triggered by a timeout:
 - $\sigma(e) = \sigma(t)$ - the timeout event occurs when the current time reaches the value of the timeout.
 - If $m \neq m_o$ (the transition sends a message), then $\sigma'(C) = \sigma(C) \cup \{\langle m, \sigma(t) + d \rangle\}$ for some $d \in I$. Else $\sigma'(C) = \sigma(C)$.
- If $e \in M$ - the transition is triggered by a message:
 - $\langle e, \sigma(t) \rangle \in Ev_{\sigma(t)}(\sigma(C))$ - the calendar has the message scheduled to occur in the current time.
 - If $m \neq m_o$ (the transition sends a message), then $\sigma'(C) = \sigma(C) \setminus \{\langle e, \sigma(t) \rangle\} \cup \{\langle m, \sigma(t) + d \rangle\}$ for some $d \in I$. Else $\sigma'(C) = \sigma(C) \setminus \{\langle e, \sigma(t) \rangle\}$.

For modeling purposes, it is convenient to describe systems compositionally. For that we also defined the composition of calendar systems, which is basically the interleaving of two systems.

4 Model Checking of Calendar Systems

A useful structural feature of calendar automata is the property that time never exceeds any time value of scheduled events [6]. Our formalism preserves this property with respect to values of currently valid timeouts and calendar events. Other invariants like the minimal and maximal value of events relative to time at a given control location or possible elements of the set of valid timeouts at a given

control location can be automatically determined by processing a graph induced by the calendar system. In the following we present our achievements in the verification of calendar systems.

4.1 Finding Counterexamples for ω -Regular Properties

As described before, model checking of an ω -regular property can be solved by searching for lassos in the product system of the original system and the automaton representing the negated property. Since calendar systems have a dense-time semantics with a monotonically increasing time variable (thus resulting in a continuous, infinite state space), in order to find lassos in the system $TS(CS) \otimes \bar{A}$ one needs a suitable bisimulation over states of $TS(CS)$. Our solution was to partition states by the time value of their scheduled events relative current time. Formally two states $\langle \ell, \sigma \rangle, \langle \ell', \sigma' \rangle$ would be equivalent if $\ell = \ell'$ and

- for all timeouts $x \in T$: $x \in \sigma(T_{valid})$ if and only if $x \in \sigma'(T_{valid})$
- for all valid timeouts $x \in \sigma(T_{valid})$: $\sigma(x) - \sigma(t) = \sigma'(x) - \sigma'(t)$
- there exists a bijection $\pi: \sigma(C) \rightarrow \sigma'(C)$ such that for each $\langle m, t_m \rangle$ where $\pi(\langle m, t_m \rangle) = \langle m', t'_m \rangle$: $m = m'$ and $t_m - \sigma(t) = t'_m - \sigma'(t)$

Although the quotient state space that can be produced with this bisimulation is still not finite, it contains lasso-shaped runs that can be recognized on the fly. This can be done by a *synchronous observer* of the system that nondeterministically saves the current state and compares each following state to that saved state [4]. If the two are equal with regard to the bisimulation relation, they are the intersection of an (abstract) lasso-shaped run.

However, this bisimulation is not necessarily coarse enough to find each such trace of the calendar system, so the method is only capable for finding counterexamples. The problem is a manifestation of the one presented in [7] for timed automata, so the evidence for this statement can be constructed analogously as the example presented there.

4.2 Proving ω -Regular Properties Using k -Induction

Proving ω -regular properties (including liveness properties) of calendar systems with k -induction can also be attempted by constructing the product $TS(CS) \otimes \bar{A}$. To prove that the number of accepting states occurring in every run of the product system is finite, one can try to find an upper bound l for the number of accepting states of a run. If such number exists, the property must hold. The method is complete for finite systems: if the property holds then there is an upper bound [5].

Although the semantics of a calendar system is not finite, for each calendar system there exists a finite system that is bisimilar to it. This statement can be proven

by giving such a bisimulation relation. Since calendar systems are very similar to timed automata (as by scheduling events only intervals bounded by natural numbers are allowed), the region equivalence [1] can be applied for this purpose. The only considerable difference would be that instead of clock values the time value of events relative to current time would be taken into account, and a clock in the unbounded clock region would correspond to an invalid timeout. As a consequence, for each calendar system, for that a ω -regular property holds, exists a suitable upper bound, namely any upper bound of its finite counterpart. As conclusion, our method can be considered complete just like in the finite case.

The existence of such an upper bound can easily be stated as an invariant property over a modified system: one must expand the system with a *synchronous observer* that counts the accepting states during the run. The property is then that the value of this counter is not greater than the upper bound. The formulated invariant property then can be checked with k-induction.

For successful verification, in our framework we support the model checker with the following settings:

- We add a supporting lemma that the value of this counter is positive (otherwise counterexamples to induction of arbitrary length could be constructed, starting from an adequately small negative counter value).
- By static analysis of the model, we provide invariants that describe the connection between the states of $TS(CS)$ and \bar{A} in the product system, by that sorting out a significant number of unreachable states.
- We set the induction depth k to be at least equal to the upper bound l , or else no counterexample during the base case can be found, since the length of paths would be too small for the number of accepting states to exceed the bound. Moreover, if the bound is greater than the induction depth, then no path in the state space will contradict the lemma over the counter values during the induction step, serving as a possible counterexample (if not sorted out by other lemmas), thus enforcing the increasing of the induction depth.

5 Tool Support

For efficient verification of calendar systems, we developed a toolchain that supports the aforementioned modeling and verification steps. Our implementation is based on the Eclipse Modeling Framework (EMF) and related technologies. It includes a domain specific language (DSL) that enables the modular description of calendar systems and the formulation of their requirements. Its metamodel is constructed in EMF and is augmented with a graphical concrete syntax that enables marking control locations and transitions between them, labeled with events (like receiving a message or that a timeout is over) and actions (like sending a message or setting a timeout). The static analysis of models focused on recognizing possible design flaws like incomplete or nondeterministic transition description or unreachable control locations. To support k-induction verification we implemented

the means for deriving the kind of invariants described in Section 4. Invariants are specified as graph patterns that can be matched over the models using the incremental graph pattern matcher EMF IncQuery [2].

For model checking, we implemented a code generator that automatically provides the mapping to the lower level artifacts that are used for model checking in the SAL environment [8]:

- The modular description of a *transition system* that corresponds to the formal semantics of the calendar system given in the instance model.
- The description of *Büchi-automata* belonging to the requirements, that can be synchronously composed with the system to provide the product system.
- The tools for finding counterexamples: an *observer for the bisimulation* and an *observer for finding loops*.
- The tools for proving properties: the *counter module for proving properties* and the derived *invariant properties*.

6 Case Study: Verification of the Industrial Protocol

Using the methods and tool described here, we managed to formally verify liveness properties of the communication protocol mentioned in Section 1. During our work, we examined the part of the protocol that establishes connections between modules and transmission of their states. We created a model of the fault-free system as a product of two calendar systems (presented in a graphical syntax in Fig. 1) that represent the field and control modules that attempt to build a connection. We then proved the liveness properties with the abstraction method described in [6], providing the abstraction in our framework as an ω -regular safety property and some lemmas corresponding to the Büchi-automaton describing it.

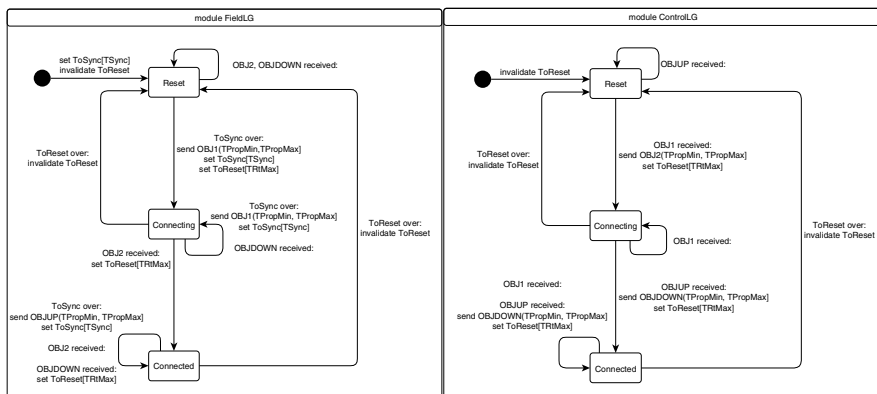


Fig. 1 Calendar system model of the field and control modules

Afterwards we extended the model with a fault model that represents losing a single message during communication. With the approach described above, we were able to find a counterexample against the liveness of the protocol: in case of specific timing conditions, loss of a single message could cause the stuck of the modules in a given state without the possibility of further progress. With the information extracted from the counterexample, we corrected the protocol and proved its correctness with respect to an unreliable communication channel where message delay may be high or messages may be lost.

7 Conclusion

The main results of our research are (1) the extension of calendar automata to provide the calendar system formalism that allows convenient modeling of the core protocols of communicating real-time systems, (2) the extension of k-induction based techniques to support the verification of both safety and liveness properties of calendar systems, and (3) the tool support to perform static analysis, derivation of invariants and artifacts required for k-induction based automated verification. The framework proved to be useful to find problems in industrial protocols. Our next steps will include the integration of other verification algorithms as well.

References

- [1] Alur, R., Dill, D.L.: A theory of timed automata. *Theoretical Computer Science* 126, 183–235 (1994)
- [2] Bergmann, G., Horváth, Á., Ráth, I., Varró, D., Balogh, A., Balogh, Z., Ökrös, A.: Incremental Evaluation of Model Queries over EMF Models. In: Petriu, D.C., Rouquette, N., Haugen, Ø. (eds.) *MODELS 2010, Part I. LNCS*, vol. 6394, pp. 76–90. Springer, Heidelberg (2010)
- [3] Biere, A., Cimatti, A., Clarke, E., Zhu, Y.: Symbolic Model Checking without BDDs. In: Cleaveland, W.R. (ed.) *TACAS 1999. LNCS*, vol. 1579, p. 193. Springer, Heidelberg (1999)
- [4] Biere, A., Artho, C., Schuppan, V.: Liveness Checking as Safety Checking. In: *FMICS 2002* (2002)
- [5] Claessen, K., Sorensson, N.: A Liveness Checking Algorithm That Counts. In: *FMCAD 2012* (2012)
- [6] Dutertre, B., Sorea, M.: Modeling and Verification of a Fault-Tolerant Real-time Startup Protocol using Calendar Automata. In: *FORMATS-FTRTFT 2004* (2004)
- [7] Kindermann, R., Junttila, T., Niemelä, I.: Complete SMT-Based Bounded Model Checking for Timed Automata. In: *FMOODS/FORTE 2012* (2012)
- [8] de Moura, L., Owre, S., Shankar, N.: *The SAL Language Manual. CSL Technical Report SRI-CSL-01-02* (2003)
- [9] Pike, L.: *Real-Time System Verification by k-Induction. NASA Technical Memorandum TM-2005-213751* (2005)
- [10] Sheeran, M., Singh, S., Stålmarck, G.: Checking Safety Properties Using Induction and a SAT-Solver. In: Johnson, S.D., Hunt Jr., W.A. (eds.) *FMCAD 2000. LNCS*, vol. 1954, pp. 108–125. Springer, Heidelberg (2000)
- [11] Vardi, M.Y.: An automata-theoretic approach to linear temporal logic. In: Moller, F., Birtwistle, G. (eds.) *Logics for Concurrency. LNCS*, vol. 1043, pp. 238–266. Springer, Heidelberg (1996)

Native Support for Modbus RTU Protocol in Snort Intrusion Detection System

Wojciech Tylman

Department of Microelectronics and Computer Science,
Łódź University of Technology,
ul. Wólczajska 221/223, 90-924 Łódź, Poland
tyl@dmc.s.pl

Abstract. The paper addresses the problem of intrusion detection in industrial networks. A novel approach to processing non-IP protocols in Snort Intrusion Detection System is presented, based on Snort Data Acquisition Module (DAQ). An example implementation for industry-standard Modbus RTU protocol is presented, which allows Snort to natively process Modbus RTU frames, without need to use external programs or hardware and without modification of Snort code. The structure of implementation and frame processing path is outlined. The solution is compared against existing attempts to process Modbus family protocols in Snort IDS. Results of tests in an virtualised environment are given, together with indications of future work.

1 Introduction

Detection of intrusions in computer networks has been subject to extensive research for many years. As can be seen in the literature of the topic, today's intrusion detection research focuses on solutions for the Internet. However, there are networks besides Internet; a very important class of these are industrial networks. Such networks, when compared with Internet on the basis of number of users may seem insignificant, but this is not the case. The industrial networks are used in environments where malfunction, or malicious activity, may result in infrastructure damage, human health hazards or even casualties.

The solutions used for Internet intrusion detection usually cannot be directly applied in the industrial environment. Industrial networks use a number of protocols never employed in the Internet, and for this reason a need arises for different capabilities. Whenever possible, the Internet solutions should be adapted for industrial needs as this allows to use a broad range of advanced and well-tested programs, many of which are also free of charge.

This paper presents an example of such approach: tailoring Snort IDS/IPS (Intrusion Detection System, Intrusion Prevention System), an open-source software that is a de-facto standard in the freeware Internet intrusion detection, to the requirements of the Modbus RTU (Remote Terminal Unit), one of the most widely

used non-IP based industrial protocol. Once Snort is capable of processing Modbus RTU traffic, it is possible to use its rule language for processing traffic in Modbus-based industrial environment, it is also possible to design custom pre-processors tailored for the specific needs of industrial networks. The discussion of the need of intrusion detection in industrial networks is beyond the scope of this paper, the reader is referred to [1] for justification of such need.

The remainders of this paper is organised as follows: Section 2 outlines Modbus RTU protocol and Section 3 introduces Snort, including the DAQ (Data Acquisition) library. Section 4 presents available solutions for processing Modbus traffic in Snort IDS. Section 5 presents the Modbus RTU DAQ module incorporating Modbus RTU support into Snort and Section 6 contains closing remarks and indications of possible further development paths.

2 Modbus RTU

Modbus protocol [2] was introduced by Modicon in 1979. Modbus RTU is its most common version, utilising in the physical layer serial communication, typically the EIA-485 standard. It allows communication between up to 240 devices; one device in the network is assigned the role of the master and only this device can initiate communication. The employed topology means that any transmission in the network is visible to all devices, but only the device the request is intended for replies. The transmission is organised into simple frames, each containing the device address (target address in case of request, source address in case of reply), function code indicating the operation performed by the slave, data related to the operation, and a cyclic redundancy check checksum. The maximum length of the frame is 256 bytes. A sample Modbus RTU frame is presented in Fig. 1.



Fig. 1 Modbus RTU frame; $t_{3.5}$ is the minimum time separating adjacent frames

The transmission of the frames over the serial link is governed by additional requirements of inter-byte and inter-frame time intervals: the bytes in frames have to be no more than the inter-byte interval apart, and the whole frames have to be more than the inter-frame interval apart. The intervals are differently calculated for transmission speeds equal or lower than 19200 bps and for speeds exceeding this value. In the former case they are equal to the transmission times of 1.5 byte and 3.5 byte, respectively. Due to this definition, the intervals are referred to as $t_{1.5}$ and $t_{3.5}$. In the latter case they are not dependent on the transmission speed and are equal to 750 μ s and 1.75 ms, respectively.

Due to its simplicity, robustness, openness and long tradition of use Modbus RTU is one of the most commonly used industrial protocols. Although many other protocols have been designed, better suited for novel, more demanding applications, most Programmable Logic Controllers (PLCs) implement this protocol; in simpler devices it is often the only protocol available.

As clearly visible from this description, Modbus RTU protocol has nothing in common with the TCP/IP protocol suite used in the Internet. Although at least two variations of Modbus are available that make use of the TCP/IP protocol suite (Modbus TCP and Modbus over TCP), Modbus RTU, due to completely different physical layer, cannot be directly processed by TCP/IP based solutions.

3 Snort IDS/IPS

Snort is an open source network intrusion detection and prevention system, initiated in 1998 by Martin Roesch, currently developed by Sourcefire [3]. It tries to detect attempts of attacks and actual attacks solely on the basis of network traffic in the monitored subnet.

An important feature of Snort, added in version 2.9, is Data Acquisition Library (DAQ). Prior to this version Snort used direct calls to PCAP library, which performs frame capturing. DAQ introduces an additional abstraction layer, making Snort itself independent of the actual implementation of the capturing layer. This feature enabled the approach described in this chapter. More information about DAQ can be found in [4].

4 Available Solutions for Processing Modbus Traffic in Snort

As far as processing Modbus RTU traffic in Snort goes, one approach has been mentioned in literature and will be presented further in this section. On the other hand, Snort itself is capable – starting from version 2.9.2 – to process Modbus and DNP3 protocols through a set of preprocessors. The solutions present in Snort are similar to those presented earlier in the Digital Bond’s Quickdraw SCADA IDS project [5]; however, unlike the latter they do not require patching the core Snort code.

It has to be stressed that both Snort and Quickdraw preprocessors are *not* capable of processing Modbus RTU. They process *only* Modbus TCP, which is the Modbus protocol positioned on top of the TCP protocol.

The approach enabling processing the Modbus RTU protocol has been described in [6] and [7]. It uses an external device (computer with appropriate software) to convert Modbus RTU protocol to Modbus TCP. The resulting TCP/IP frames are then passed to Quickdraw SCADA IDS, which performs actual detection. The solution works as an IDS, but an IPS configuration is also possible with

separate receive and transmit modules and Snort placed in between. However, the latter introduces additional delays in serial transmission.

The advantage of the solution is support for non-IP protocols – currently Modbus RTU and Modbus ASCII. Moreover, this approach does not require modifications of Snort.

The disadvantage is a cumbersome installation – the authors propose a configuration with two virtual computers, connected by a virtual Ethernet link, both of which capture Modbus RTU traffic; Snort IDS is installed on one of these virtual machines and processes both TCP/IP streams resulting from conversion of Modbus RTU to Modbus TCP. Moreover, as the conversion module is separate from the Snort itself, it is difficult to incorporate some advanced functionality – for example, the IPS configuration presented by the authors suffers from additional delays.

5 Modbus RTU DAQ

The aim of this work was to enable processing of Modbus RTU in Snort without the need of external programs. The introduction of DAQ allowed to achieve this goal without any modification of the Snort code: the new DAQ module is an addition, not a modification.

5.1 *Structure of the Modbus RTU DAQ*

Every Snort DAQ module is an intermediate layer placed between actual Snort and the code responsible for acquisition of frames in the network (acquisition code, as it will be subsequently called). The interface between Snort and DAQ is standardised by DAQ API, the interface between DAQ and acquisition code is strictly dependent on the latter. The detailed discussion of the Snort DAQ API is beyond the scope of this chapter. For the purpose of the discourse it is enough to outline the overall concept of this solution while presenting the structure of Modbus RTU DAQ.

Most important DAQ functions belong to two groups: a) initialisation, and b) acquisition process. Of these, the initialisation phase is responsible for configuring the acquisition module, optionally using the arguments passed in the command line of Snort. DAQ uses a flexible method based on a collection of key-value pairs, where both keys and values are character strings. In case of Modbus RTU DAQ it allows to pass serial port name and its parameters.

The acquisition process is outlined in Fig. 2. It is started by calling a single function, by convention named `x_daq_acquire`, where 'x' stands for the name of the particular DAQ module. The function arguments at least specify the callback which is the entry point to the detection procedure in Snort core and the number of frames to process. This function calls a function in the acquisition code, which is

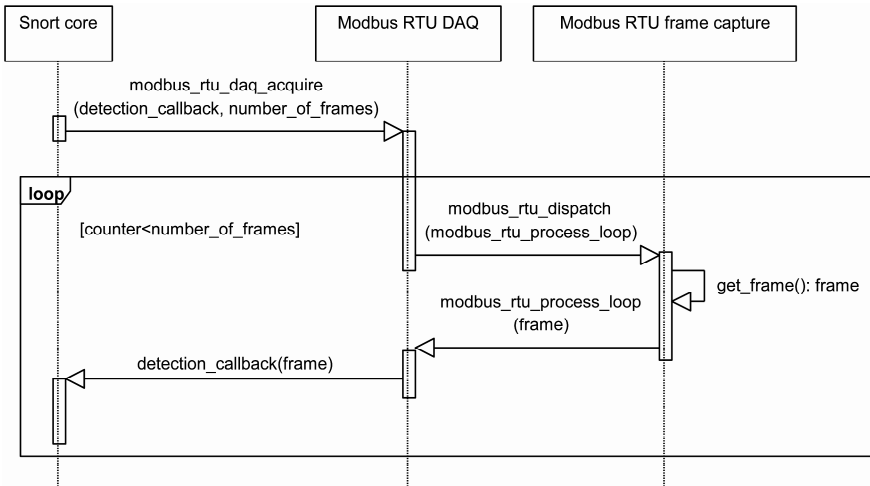


Fig. 2 Outline of frame acquisition with Modbus RTU DAQ

responsible for a) retrieving a frame from the network, and b) calling an user supplied callback function with the retrieved frame as an argument. The mentioned callback is usually also a part of DAQ; ultimately this call results in performing, for each frame, the whole detection procedure inside Snort core.

5.2 Capturing Modbus RTU Frames

Capturing Modbus RTU frames involves reading data from the serial port. This is essentially a straightforward task, as the operating system provides appropriate functions. However, Modbus RTU protocol impose timing requirements, as described in Section 2. This means the bytes have to be read one by one and the intervals between bytes calculated and compared against specification. It also means the whole processing of the data must fit in the $t_{3.5-t_{1.5}}$ interval (assuming only complete frames are processed).

This requirement should not be a problem for a computer wholly dedicated to processing Modbus RTU traffic in a single network. Snort IDS is capable of processing TCP/IP traffic of 300 Mb/s on a single processor computer [8]; this translates to processing time of less than 30 μ s per frame for 1 kB frames, well below the minimum $t_{1.5}$ interval of 750 μ s. However, if a single computer is to monitor several networks, perhaps including a more heavily loaded TCP/IP network, the timing requirements may become a problem. For this reason Modbus RTU DAQ uses a separate thread, devoted only to capturing the frames over the serial line. The captured frames are stored in a FIFO queue and from there consumed by Snort. In such approach Snort may process frame in time equal to transmission

time of the next frame. Assuming shortest valid Modbus RTU frame (4 bytes), this gives over 2 ms for 19200 bps link. Even if, for some reason, the processing takes longer, the frames are not lost due to the queue concept.

5.3 *Feeding Modbus RTU Frames into Snort*

Once the frames are captured, they have to be processed by Snort intrusion detection algorithm. However, Snort is expecting frames coming from the second layer of the TCP/IP stack, which can be Ethernet, Wi-Fi, etc. These formats are not compatible with Modbus RTU frame. Moreover, further processing is also based on the TCP/IP stack, meaning Snort is decoding IP datagrams, TCP packets, etc. These obviously are not present in the Modbus RTU frame. In order to solve this problem, one of two solutions may be applied.

The first one is to modify the Snort source code to include an additional path for processing new type of frames. This solution is tempting, as it allows to address specific aspects of the protocol (e.g., analyse flags, perform session reassembly, etc.). On the other hand, it requires quite substantial modification of the Snort code: although much of processing specific to, for example, TCP protocol, is performed in preprocessors, which can be easily added ore removed (in a manner similar to DAQs), significant portion of the current processing path is contained in the core of Snort. Consequently, writing only a preprocessor to handle a new protocol is possible if this protocol is based on the lower layers (second and third) of the TCP/IP stack – in other cases modification of the Snort core is required.

The second approach is to wrap the new protocol frame so that it is contained in the payload part of one of the protocols supported by Snort. As an example, an Ethernet frame with an IP datagram and an UDP packet can be built. Such frame may then be directly processes by unmodified Snort. The drawback of this approach is the fact that usually the only detection approach that can be applied to the frame is analysis of the packet payload, as all other parts of the frame are fake. Fortunately, payload analysis is the most powerful element of Snort, so it should be sufficient in case of simple protocols. Sometimes it is also possible to map parts of the payload to the fields of the enclosing protocol frame in order to make better use of the rule language of Snort.

As Modbus RTU is a simple protocol, the second approach may be used. The frames passed from DAQ to Snort are therefore fake Ethernet frames with Modbus RTU frame as UDP payload. To facilitate efficient use of Snort rule syntax, two elements of the Modbus RTU frame are copied to the fields of the IP datagram and UDP datagram: slave address is copied to the least significant byte of the destination IP address and Modbus function code is copied to the destination UDP port number. The processing path of the Modbus RTU frame is depicted in Fig. 3.

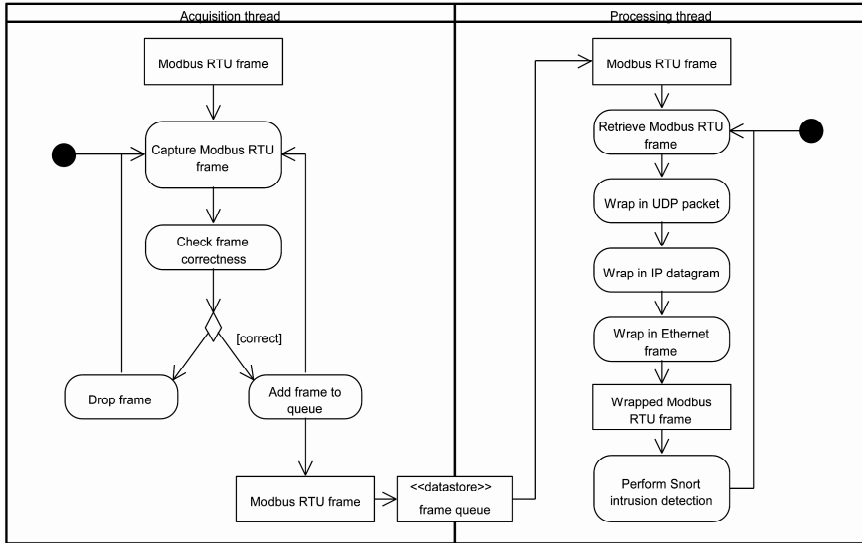


Fig. 3 Modbus RTU frame processing path

5.4 Tests of the Solution

The solution has been tested using virtualisation techniques. A set of interconnected virtual serial ports has been created, to which simulators of Modbus RTU master and Modbus RTU slaves have been connected. One of the ports was used by Snort. Snort configuration has been altered to disable its preprocessors and rely only on the rules for payload analysis. A sample set of rules triggering at the commonly used Modbus functions or specific data patterns has been created. The solution has been tested on uniprocessor PC computer running Microsoft Windows 7 Professional 32-bit operating system.

The test proved correctness of the solution. Modbus RTU frames transferred between the master and the slaves were correctly captured by the Modbus RTU DAQ and, after placing inside of an UDP packet, processed by Snort. In case of frames which elements matched the sample rules, Snort generated alarms. Other frames did not cause any activity.

Figure 4 presents a sample Snort rule, detecting “Write single coil” command, addressed at slave number 4 or 6, and requesting to write coil (output) of number equal to 10 or greater. The sample frame matching this rule is also shown.

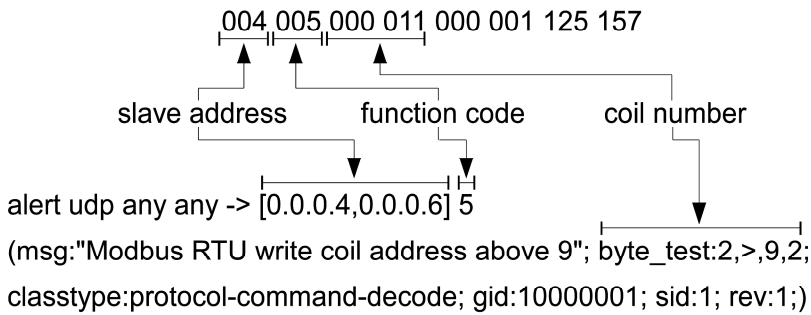


Fig. 4 Sample Snort rule for Modbus RTU and a frame matching this rule

6 Conclusions and Future Work

This paper presented a novel approach to the intrusion detection with Snort IDS in non-IP networks. It demonstrated the possibility to add native support for non-IP protocols by using Snort DAQ, an intermediate layer designed as an interface between the frame capturing code and the Snort core. The tests proved the solution is capable of detecting predefined data patterns in Modbus RTU frames using Snort rules.

There are many possible ways to expand the presented solution. First of all, Modbus RTU is only one of many protocols used in industry. Other non-IP protocols, such as for example Profibus or Foundation Fieldbus are also widely used and support for them should be added. Another consideration is assuring compatibility with the Snort Modbus preprocessor, described in Sect. 4.1, which was not available at the time the solution presented here was developed. This should not be a problem, as Modbus TCP is similar in its concept to the idea used in the presented solution, i.e., of placing Modbus RTU frame inside of an Ethernet frame.

It should also be noted that the concept presented in this paper is a building block of a larger project, aiming at providing SCADA intrusion detection solution, based on Snort and elements of artificial intelligence, mainly Bayesian networks. The project will incorporate results obtained in [9].

Acknowledgements. The work described in this chapter has been financed by the National Science Centre (agreement No. 4769/B/T02/2011/40).

References

- [1] Tylman, W.: SCADA Intrusion Detection Based on Modelling of Allowed Communication Patterns. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *New Results in Dependability & Comput. Syst.* AISC, vol. 224, pp. 489–500. Springer, Heidelberg (2013)
- [2] Modbus RTU specification (2012), <http://www.modbus.com> (accessed February 2012)

- [3] Snort home page (2012), <http://www.snort.org> (accessed February 2012)
- [4] Snort DAQ description (2012), <http://vrt-blog.snort.org/2010/08/snort-29-essentials-daq.html> (accessed February 2012)
- [5] Digital Bond Quickdraw SCADA IDS (2010), <http://www.digitalbond.com/tools/quickdraw> (accessed February 2012)
- [6] Morris, T., Pavurapu, K.: A retrofit network transaction data logger and intrusion detection system for transmission and distribution substations. In: Proc. IEEE Int. Conf. Power and Energy (PECon), Kuala Lumpur, Malaysia, pp. 958–963 (2010)
- [7] Morris, T., Vaughn, R., Dandass, Y.: A Retrofit Network Intrusion Detection System for MODBUS RTU and ASCII Industrial Control Systems. In: Proc. Int. Conf. System Sciences, Maui, Hawaii, USA, pp. 2338–2345 (2010)
- [8] Graham, I.: Achieving Zero-loss Multi-gigabit IDS Results from Testing Snort on Endace Accelerated Multi-CPU Platforms (2012), <http://www.touchbriefings.com/pdf/2259/graham.pdf> (accessed February 2012)
- [9] Tylman, W.: Detecting Computer Intrusions with Bayesian Networks. In: Corchado, E., Yin, H. (eds.) IDEAL 2009. LNCS, vol. 5788, pp. 82–91. Springer, Heidelberg (2009)

SCADA Intrusion Detection Based on Modelling of Allowed Communication Patterns

Wojciech Tylman

Department of Microelectronics and Computer Science,
Łódź University of Technology,
ul. Wólczańska 221/223, 90-924 Łódź, Poland
tyl@dmc.s.pl

Abstract. This work presents a network intrusion detection system (NIDS) for SCADA developed as an extension to Snort NIDS, a popular open-source solution targeted at intrusion detection in Internet. The concept of anomaly-based intrusion detection and its applicability in the specific situation of industrial network traffic is discussed. The idea of modelling allowed communication patterns for Modbus RTU protocol is explained and the system concept, utilising n-gram analysis of packet contents, statistical analysis of selected packet features and a Bayesian Network as data fusion component is presented. The implementation details are outlined, including the concept of building the system as a preprocessor for the Snort NIDS. The chapter is concluded by results of test conducted in simulated environment.

1 Introduction

Intrusion detection based on analysis of network traffic is a well-known concept and many research efforts have been targeted at improving its accuracy. Discovering malicious activity on the basis of data passed in a network is an obvious approach, considering the fact that nowadays most of attacks are *performed* using computer networks. Analysis of both currently used solutions and scientific literature shows that this concept almost exclusively focuses on intrusion detection in Internet and network types commonly participating in carrying Internet traffic. This is not surprising, considering the ubiquitousness of Internet; nevertheless, there are other network types besides these associated with Internet. The question arises: is network intrusion detection needed and feasible in such networks? And again: are the concepts developed for Internet best suited to such networks?

One category of such networks are the so-called industrial networks. This term encompasses a wide variety of networks, differing in topology, physical layer properties, complexity of network protocol stack, etc. They are used not only in industrial establishments (e.g., factories), but also in vehicles and aircraft; applications include also other scenarios where robustness of operation is required. Typical role of such networks is to connect Programmable Logic Controllers (PLCs)

with other equipment, notably with a PC computer acting as a supervisory system, thus creating communication infrastructure for Supervisory Control And Data Acquisition (SCADA). Such networks, for reasons explained further in this work, have long been viewed as immune to attacks. Recent events have shown it not to be the case and should be a clear indication that solutions are needed to protect industrial networks as well.

The remainder of this work is organised as follows: Section 2 discusses current state of intrusion detection for industrial networks and SCADA systems. Section 3 outlines Modbus RTU, a simple yet commonly used type of industrial network protocol that serves as a proving ground for concepts described in this work. Section 4 introduces Snort NIDS and its concept of pre-processors, used to implement the presented solution. Section 5 offers details of the developed approach to intrusion detection. Section 6 presents results of tests, while Section 7 contains closing thoughts and outline of future work.

2 Intrusion Detection in Industrial Networks and SCADA

As already mentioned, intrusion detection in industrial networks is a subject that has not yet drawn too much attention, especially when compared to intrusion detection in Internet. There are a number of historical reasons that led to such situation:

- **isolation of industrial networks** – early networked control systems were physically separated from outside world, therefore it was not possible to introduce harmful communication in any other way than through *physical* presence of the attacker inside the industrial establishment,
- **non-routable protocols** – even when the industrial networks become connected to other networks (typically through the SCADA PC, which could be connected both to the control system networks and standard TCP/IP based office networks) it was believed that the control system could not be attacked, as the protocols used in the industrial networks were not routable, i.e. the data could not be directly passed between the industrial network and the TCP/IP-based networks,
- **security by obscurity** – industrial control systems use highly-specialised hardware that, unlike PCs, is of a very limited application. Moreover, early control systems used custom hardware and software solutions, which further limited the group of people possessing knowledge needed for successful exploitation of potential vulnerabilities,
- **simplicity** – although microprocessor-based, PLCs were traditionally much less sophisticated than contemporary PC hardware and software. This led to an assumption that their software lacked the necessary capabilities that could be tampered with to perform an attack.

Unfortunately, all these assumptions are currently unfounded: control systems are connected to Internet, they use of-the-shelf components which operation is

explained in undergraduate courses and the hardware capabilities of current PLCs often exceed those of the still-used office PCs. As to the question of non-routable protocols, recent events, such as the attacks performed using the Stuxnet worm [1], proved that non-routable protocols are by no means a barrier to successful exploitation.

The deceptive faith in immunity of industrial networks successfully crippled potential research: only a very limited number of research projects targeting intrusion detection in industrial networks emerged. Among these notable exceptions are:

- a project sponsored by the U.S. Department of Homeland Security showing the possibility to use models of communication patterns as a base for delimiting malicious traffic ([2], [3], with further developments presented in [4]),
- the VIKING project financed through EU 7th Framework Programme, primarily concerning security in power delivery systems [5].

Neither of these projects resulted in practical application. One possible reason is the fact that the proposed solutions were not compatible with existing and popular Internet NIDS systems – therefore they could not take advantage of existing extensions, knowledge base, support, etc. of such systems. This problem was noticed by the Digital Bond company, who developed Quickdraw SCADA IDS [6], building on the foundation of Snort, a de-facto standard for open-source Internet NIDS. However, this solution is no more than a set of pre-processors for Snort that allow it to process data from selected IP-based industrial networks – there are no new detection methods tailored for industrial networks, nor support for non-IP industrial networks. Still, its concept of extending Snort using the pre-processor mechanism became inspiration for the work described in this chapter.

3 Modbus RTU

Modbus RTU (Remote Terminal Unit) is a simple master-slave protocol developed by Modicon in 1979. It operates over a serial line (typically the EIA-485 standard). Modbus RTU can be treated as an application layer protocol (OSI model layer 7) with EIA-485 serving as a physical layer (OSI model layer 1). Other layers in the OSI model are null for Modbus RTU. The Modbus RTU specification [7] defines communication frame (Fig. 1), which is the basic unit of data transmission. Depending on the value of a *function code* field in the frame, various operations are possible, most of which concern reading or writing inputs, outputs and registers of the PLC. Up to 240 devices can be connected, of which only one performs the role of a master. Due to its simplicity and maturity it is one of the most commonly used protocols in industrial networks.

Modbus RTU has been chosen as the proving ground for concepts presented in this work not only because of its popularity, but also because it is completely different (both in physical and logical aspects) from the TCP/IP stack protocols that are used by Snort.

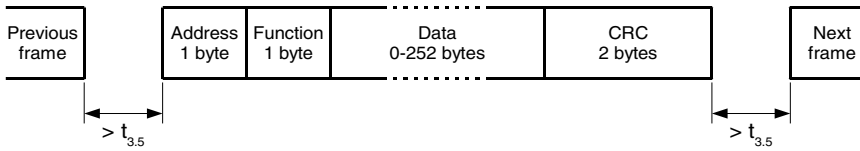


Fig. 1 Modbus RTU frame; $t_{3.5}$ is the minimum time separating adjacent frames

4 Snort NIDS

Snort [8] is an open-source NIDS first released in 1998 and currently developed by Sourcefire. Its principle of operation is based mainly on misuse detection, i.e. comparing the traffic against a set of rules describing forbidden patterns (in majority these are contents of packets known to occur when attacks are in progress). This approach (although conceptually simple, lacking the capability to self-adjust to observed traffic and requiring a constant supply of human-created rules) proved most reliable for Internet intrusion detection.

Snort has been chosen as a base for the presented solution not only because of its widespread use, but also because of its flexible system of pre-processors and Data Acquisition Libraries (DAQs). The concept of preprocessors allows to easily extend Snort detection capabilities with arbitrary code, while DAQs, introduced in version 2.9, allow to provide custom modules for capturing data. Such module is required in the discussed solution, as Snort by itself is only capable of capturing data in IP-based networks.

5 The Proposed Concept

The concept presented in this work follows the principles of anomaly detection. Unlike misuse detection, which *blacklists* malicious traffic, anomaly detection attempts to *whitelist* allowed traffic: it tries to capture some characteristics of normal traffic and alerts if the observed traffic deviates from this norm. This inherently leads to a two-phase operation: in the first, learning, phase the detector only observes the traffic, in the second, detection, phase it pronounces decisions. In practice the learning phase is periodically repeated in order to adjust the detector to the changing operation pattern of the protected system.

Although conceptually tempting due to fully automatic operation (no human-created rules, self-adjustment to the observed traffic) this concept proved unreliable for Internet NIDSes due to complexity of Internet network traffic and variability of human user behaviour. However, situation is significantly different in industrial networks: here the traffic is simple (usually only one protocol and a limited number of data types) and the industrial equipment, which is the “user”, displays clear patterns of “behaviour”. For these reasons, anomaly detection should be feasible in case of industrial systems.

It should be noted that the term *anomaly* is a broad concept: what is an anomaly in one case, might be a perfectly valid behaviour in another case. Moreover, anomalous behaviour does not necessarily must be related to an intrusion: a transmission attempting to set PLCs output to values well beyond normal operating range may be an indication of an attack, but it may also indicate fault with the supervisory algorithm. However, in both cases it should be detected.

5.1 Structure of the Detector

The general structure of the proposed detector is schematically presented in Fig. 2. Data are sensed on the serial line (EIA-485) using a tap (in this case a serial port in receive mode). The data are captured by a DAQ library specially prepared for Modbus RTU (see Section 5.2). Further processing is performed in the preprocessor, which computes a number of characteristic values and fuses them using a Bayesian Network (see Section 5.3). The preprocessor constitutes in fact the core of the detector: all processing leading to anomaly detection is performed inside this preprocessor and the decisions about packets are passed to the main part of Snort. However, as the data captured by the DAQ library are also made available to the further processing stages of Snort, it is possible to use the Snort rule language to explicitly define forbidden traffic, thus creating a hybrid, misuse-anomaly solution.

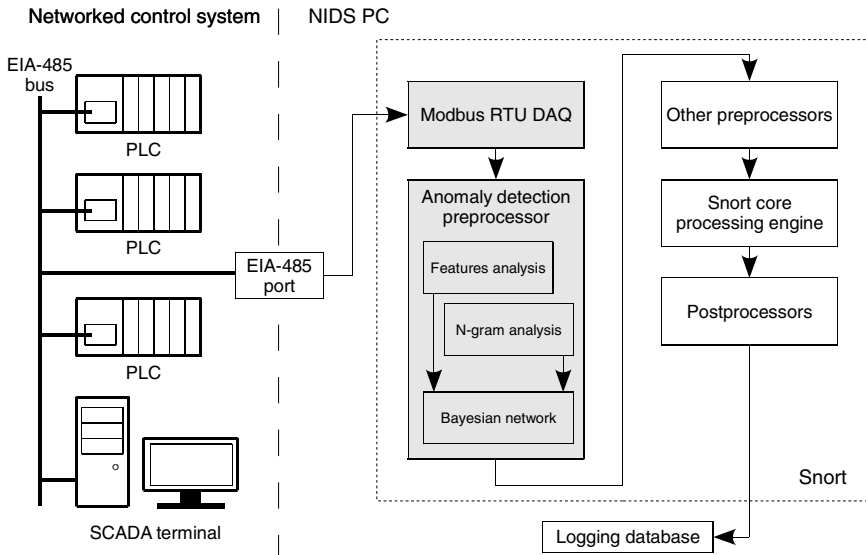


Fig. 2 Overview of detection system. The elements developed in this work are on gray background.

The last stage of processing are the standard Snort post-processors, allowing operations such as logging events (e.g., detected malicious activities) to databases.

5.2 *DAQ Library for Modbus RTU*

The task of the library is to capture data from the serial port and feed it to subsequent processing stages of Snort NIDS. The task of reading data from the serial port is straightforward, although certain timing restrictions imposed by the Modbus RTU specification have to be obeyed in order to correctly delimit frames and reject spurious transmissions. However, the problem arises when the captured frames are to be dispatched to the further stages of Snort: Snort, in its internal processing path, is expecting data adhering to one of the protocols used in Internet. This means the data should at least be formatted in accordance with the IP protocol in OSI layer 3 and in accordance with the TCP or UDP protocol in OSI layer 4. Because Modbus RTU is a non-IP protocol, such formatting is not present in the captured data. In order to allow processing Modbus RTU data by Snort two approaches are possible:

- altering the internal processing path of Snort to add the data format of Modbus RTU,
- re-formatting the captured data in order to adjust them to the requirements of Snort.

The first of these solutions would require significant modifications of Snort, so the second solution is used: the whole captured frame (excluding the CRC fields) is placed as payload of a fake UDP datagram, which is placed as payload of a fake IP datagram and subsequently in a fake Ethernet (IEEE 802.3) frame. Moreover, slave address from the Modbus RTU frame is copied to the least significant byte of the destination IP address and the Modbus function code is copied to the destination UDP port number – these operations are performed in order to facilitate writing rules for misuse detection using Snort rule language. The frame thus created is then passed down the processing path.

More details on this concept, as well as its comparison to other solutions for processing industrial network traffic in Snort, can be found in [9].

5.3 *Detection Preprocessor*

The actual anomaly detection is performed in the preprocessor. Three distinct components can be distinguished, they will be described in details further in this section. All analyses are performed separately for any pair of endpoints (PLCs, PCs, Human-Machine Interface panels, etc.) between which communication has occurred in the learning phase. If during the detection phase a communication is seen between endpoints that never communicated in the learning phase, it is immediately tagged as anomaly.

Analysis of Packet and Conversation Features

The first component performs analysis of features without inspecting the data carried by the packet. The analyses are performed with respect to a *conversation*, i.e. a delimited sequence of packets passed between two endpoints. The way to split packets into conversations depends on the protocol used: in the TCP protocol the conversation is synonymous with session and can be processed on the basis of sequence numbers, however, in Modbus RTU no such concept is present. Therefore the conversation for the latter protocol is defined, for the purpose of the developed system, as a pair of packets, the first one originating from the master to a slave (request), the second one from the slave to the master (reply). The analysed features are following:

- size of conversation in bytes,
- size of conversation in packets,
- mean packet size in conversation,
- standard deviation of packet size in conversation,
- duration of the conversation.

Due to the above-mentioned definition of conversation in Modbus RTU, the second of these features will have a constant value of 2 (for complete conversations). However, this feature can be useful for other network types.

Analysis of Packet Payload

Analysis of packet payload is performed using the concept of n-grams. An n-gram is an n-element contiguous sequence; when applied to the network traffic data this is an n-element sequence of bytes. This kind of analysis has been proposed in [10] to detect propagation of Internet worms by comparing the data consumed by the host with the data produced by the host. In the proposed solution n-grams are used to approximate typical data passed between two endpoints by computing how often particular n-grams are occurring in the payload (an n-gram profile). As the algorithm uses unigrams (1-grams), this translates to computing how often particular *bytes* are occurring. Separate analyses are performed for different lengths of the observed frames.

The learning of the module is two-stage: in the first phase the module computes and saves an n-gram profile for every observed frame. In the second phase it is computed how much the n-gram profile for each observed frame differs from those saved in the first phase. The aim of this two-stage approach is to estimate how much the n-gram profile changes in time, and therefore how significant must be the deviations from the profile observed in the learning phase to call an alarm in the detection phase. To this end, an Euclidean distance $d(n, s)$ is computed between the profile of the new frame and each of the saved profiles. The smallest value of $d(n, s)$, $d(n, s)_{min}$ is an indication of how much the new frame differs from the most similar frame observed during the first phase. However, such information is incomplete, because it does not take into account how the saved profiles differ between themselves. This aspect is important: the system should differently treat

the cases where $d(n, s)_{min}$ is the same, but the variability between frames observed in the first phase is different. This problem is schematically depicted in Fig. 3: in situation a), the new frame should be treated as an outlier, whereas in situation b) it should not, even though $d(n, s)_{min}$ is the same.

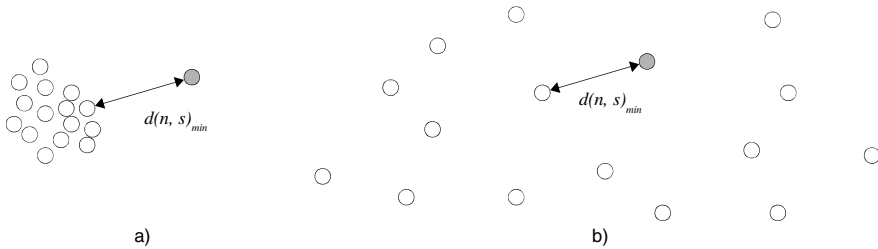


Fig. 3 Relativity of the minimum distance to the previously observed frames

The typical approach to such problem would be to perform cluster analysis on the saved profiles and use cluster parameters (for example its radius) as an indication of variability between the saved frames. However, such analysis would be troublesome, mainly because of the data high dimensionality [11] (the space has 256 dimensions, as the profile contains entries for each possible value of byte). For this reason, another approach has been used: instead of considering only one saved frame nearest to the new frame, N distinct nearest saved frames are considered and the average distance between these frames $d(s, s)_{avg}$ is the measurement of the variability. The result of the analysis is the ratio $d(n, s)_{min} / d(s, s)_{avg}$.

The n -gram analysis is performed for the whole payload present in the standard Snort packet structure. If the network protocol is Modbus RTU, this means that, due to encapsulation of the whole frame in the payload field of the UDP datagram, the analysis will also include slave address field present in the Modbus RTU frame. This field strictly speaking is not a payload, however, as it is constant for any given endpoints pair, it will not have an impact on the analysis (or, precisely speaking, its impact will be the same for all captured conversations, and therefore can be neglected).

One can argue that it is possible to perform a more detailed analysis of the payload, taking into account the structure of the Modbus RTU frame: for example it could be possible to analyse separately the typical function codes used in the transmissions. Although such approach could be easily implemented, the concept of this analysis was to be protocol-agnostic, and therefore the whole payload is treated in the same manner.

Fusion of Results

An important novelty of the proposed work lies in the concept of fusing the results of analyses, described in the previous paragraphs, utilising a Bayesian Network (BN) [12]. Although the, already cited, works concerning intrusion detection in industrial networks ([2], [5]) indicated possibility of using Bayesian approach, this

possibility has not been explored. The author of this work has already used BNs in an anomaly-based NIDS for the Internet [13]; however, in that approach the structure and most of the numerical specification of the BN were human-created. The concept presented here uses fully automatic generation of the BN based on captured data.

As the BN used is a discrete one, the data from the conversation feature analysis and the n-gram analysis have to be discretised prior to passing them to the BN; the discretisation results in variables having K states. The discretisation is performed in such way that the data gathered during the learning phase correspond to $K-2$ central bins, leaving the edge bins for possible outliers.

The important problem that had to be solved was how to perform learning of the BN having only the data representing normal (not malicious) activity. This problem of learning one-class data is common to all anomaly-based system. In the proposed detector the problem was solved by using in the learning procedure three distinct datasets:

1. a dataset generated using data gathered in the network; for all samples in this dataset the desired value of the output (decision) field was set to “no anomaly”,
2. an artificially generated dataset, in which the samples had random values of the input fields and the desired value of the output field set to “anomaly”,
3. a dataset generated as in point 2, but with the desired value of the output field set to “no anomaly”.

The datasets 2 and 3 serve as a background noise against which the learning procedure has to distinguish the real data representing normal traffic. Dataset 2 is intended to provide samples with input combinations not present in the normal traffic and instruct the learning procedure that these samples are *not* normal traffic. Dataset 3 is intended to somewhat offset the influence of dataset 2: otherwise a sample for which a single input value were outside the range observed during learning could be classified immediately as anomaly – such behaviour of the detector would be overly simplistic and identical to simple thresholding of the input values. For this approach to work the size of dataset 2 must be significantly larger than dataset 3; during evaluation of the approach dataset 3 had the same size as dataset 1, while dataset 2 was 5 times larger.

6 Preliminary Tests

Preliminary test have been conducted in a simulated networking environment. As the current concept of the system considers each pair of conversing endpoints separately – there is no support for detection of anomalous order of conversations – at this stage it is sufficient to evaluate only one pair of endpoints. Consequently, a set-up with one master, one slave and one tap has been developed using virtual serial ports software [14]. Both master and slave were also simulated in software [15], [16].

A number of scenarios simulating simple communication patterns have been tested. The aim of the tests was to evaluate how different the conversation has to be from these in the learning phase to detect an anomaly and whether the conversations identical to these in the learning phase are always correctly classified as normal traffic. It should be noted that a behaviour when *every* conversation differing from the learning phase is classified as anomaly is not a desired behaviour – instead, some differences should be permitted, and the level of this flexibility should depend on the variability of conversation pattern observed during the learning phase.

Table 1 Tests descriptions

Test No.	Learning phase conversations	Detection phase conversations
1	Request: read eight discrete inputs; input starting address constant Response: correct response; values of all inputs equal to 0	Request: as in learning phase Response: correct response; value of input 0 changing randomly, values of all other inputs equal to 0
2	Request: read eight discrete inputs; input starting address constant Response: correct response; values of inputs 0, 1 and 2 quickly changing randomly, values of all other inputs equal to 0	Request: as in learning phase Response: correct response; values of all inputs changing randomly
3	Request: read eight discrete inputs; input starting address constant Response: correct response; values of inputs slowly changing in following sequence: 00000000, 00000001, 00000011, 00000111, 00001111	Request: as in learning phase Response: correct response; values of all inputs changing randomly
4	Request: read eight discrete inputs; input starting address constant Response: correct response; values of inputs slowly changing in following sequence: 00000000, 00000001, 00000011, 00000111, 00001111	Alternating: 1) Request: as in learning phase Response: correct response; values of all inputs changing randomly 2) Request: read single analog input; input address constant Response: correct response; value of input equal to 0
5	Alternating: 1) Request: read eight discrete inputs; input starting address constant Response: correct response; values of all inputs changing randomly 2) Request: read single analog input; input address constant Response: correct response; value of input equal to 0	Alternating: 1) Request: read eight discrete inputs; input starting address constant Response: correct response; values of all inputs changing randomly 2) Request: read single analog input; input address constant Response: correct response; value of input changing with values equal to 0, 10, 30.

The test scenarios are described in Table 1, while the numerical results are presented in Table 2. The following conclusions may be derived:

- in all cases the traffic identical to that observed during the learning phase is correctly classified as normal,
- when the data transmitted in the learning phase are constant (test 1) or belong to a clearly defined set and their distribution does not change with time (test 2), even minute changes in the data observed in the detection phase cause an alarm,
- when the distribution of the data transmitted change during the learning phase (test 3), even significant changes in the data observed in the detection phase do not cause an alarm,
- the system is able to distinguish frames not observed during learning phase on the basis of their length (test 4),
- the system is able to discern and differently treat frames of different lengths (test 5).

Table 2 Tests results

Test No.	% of conversations identical to learning classified as normal	% of conversations different from learning classified as anomaly	Notes
1	100%	100%	1 % of conversations carrying the same data as the ones in the learning phase have been classified as anomaly due to significantly different conversation duration caused by glitches of the simulation environments
2	100%	100%	
3	100%	0%	
4	100%	50%	All conversations involving reading discrete inputs classified as normal, all conversations involving reading analog input classified as anomaly
5	100%	100%	

7 Conclusions and Future Work

This chapter presented a novel approach for hardening networked control systems – an anomaly-based NIDS. The preliminary tests of the solution prove the system is capable of performing detection of anomalies in transmissions between the devices communicating using the Modbus RTU protocol. The solution exhibits desired properties, i.e. is able to detect small changes in the communication patterns when the variability of these patterns during the learning phase is minimal, but allows significant deviations in case the variability during the learning phase is substantial.

The future work will include tests of the solution in a set-up consisting of typical networked control system components (PLCs connected to a PC computer using an EIA-485-based network), as well as expanding the approach through modelling of the behaviour of the devices controlled by the PLCs.

Acknowledgements. The work described in this chapter has been financed by the National Science Centre (agreement No. 4769/B/T02/2011/40). The presented solution utilises SMILE and SMILearn Bayesian network libraries, developed by Decision Systems Laboratory, University of Pittsburgh and available at <http://genie.sis.pitt.edu>.

References

- [1] Falliere, N., Murchu, L.O., Chien, E.: W32.Stuxnet Dossier. In: Symantec Security Response (2011), http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf (accessed January 19, 2013)
- [2] Cheung, S., Dutertre, B., Fong, M., Lindqvist, U., Skinner, K., Valdes, A.: Using Model-based Intrusion Detection for SCADA Networks. In: Proceedings of the SCADA Security Scientific Symposium, Miami Beach, Florida (2007)
- [3] Valdes, A., Cheung, S.: Intrusion Monitoring in Process Control Systems. In: Proceedings of the 42nd Hawaii International Conference on System Sciences, Waikoloa, Big Island, Hawaii (2009)
- [4] Valdes, A., Cheung, S.: Communication Pattern Anomaly Detection in Process Control Systems. In: Proceedings of the IEEE International Conference on Technologies for Homeland Security, Waltham, Massachusetts (2009)
- [5] Viking Project homepage (2011), <http://www.vikingproject.eu> (accessed January 14, 2013)
- [6] Digital Bond Quickdraw SCADA IDS (2010), <http://www.digitalbond.com/tools/quickdraw> (accessed January 14, 2013)
- [7] Modbus RTU specification (2012), <http://www.modbus.org/specs.php> (accessed January 14, 2013)
- [8] Snort home page (2013), <https://www.snort.org> (accessed January 14, 2013)
- [9] Tylman, W.: Native Support for Modbus RTU Protocol in Snort Intrusion Detection System. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *New Results in Dependability & Comput. Syst. AISC*, vol. 224, pp. 479–487. Springer, Heidelberg (2013)
- [10] Wang, K., Stolfo, S.J.: Anomalous Payload-Based Network Intrusion Detection. In: Jonsson, E., Valdes, A., Almgren, M. (eds.) *RAID 2004. LNCS*, vol. 3224, pp. 203–222. Springer, Heidelberg (2004)
- [11] Kriegel, H.-P., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data* 3(1), 1–58 (2009), doi:10.1145/1497577.1497578
- [12] Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Mateo (1988)
- [13] Tylman, W.: Anomaly-Based Intrusion Detection Using Bayesian Networks. In: Proceedings of the Third International Conference on Dependability of Computer Systems DepCoS-RELCOMEX 2008, Szklarska Poręba (2008)
- [14] Null-modem emulator com0com (2012), <http://com0com.sourceforge.net> (accessed January 2013)
- [15] Modpoll Modbus Master Simulator (2012), <http://www.modbusdriver.com/modpoll.html> (accessed January 14, 2013)
- [16] Modbus PLC Simulator (2011), <http://www.plcsimulator.org> (accessed January 14, 2013)

System for Estimation of Patient's State – Discussion of the Approach

Wojciech Tylman¹, Tomasz Waszyrowski², Andrzej Napieralski¹,
Marek Kamiński¹, Zbigniew Kulesza¹, Rafał Kotas¹, Paweł Marciniak¹,
Radosław Tomala¹, and Maciej Wenerski¹

¹ Department of Microelectronics and Computer Science,
Łódź University of Technology,

ul. Wólczańska 221/223, 90-924 Łódź, Poland

tyl1@dmcs.p.lodz.pl

² Norbert Barlicki University Clinical Hospital No. 1 in Łódź,
ul. Kopcińskiego 22, 90-153 Łódź, Poland

Abstract. This chapter presents work on a diagnosis support system with continuous estimation of a sudden cardiac arrest risk. The concept is based on the analysis of the signals received on line from a medical monitor, possibly augmented by clinical data. The need for such solution is motivated and existing approaches towards quantitative estimation of patient's health are examined. An overview of the system and its building blocks are presented, which allows to define the requirements for the hardware platform and operating system. A discussion follows, explaining the choice made. The design of communication path between the medical monitor decided upon and the discussed system is explained. The prototyping microprocessor board being used is presented, together with detailed description of steps which lead to the development of a fully-featured Linux distribution for this board.

1 Introduction

Cardiovascular diseases are the most common cause of death (nearly 50% – as shown in Figure 1) and hospitalizations (16% from group of 6 400 000 patients in 2006) in Poland [1]. The most common direct cause of hospital mortality is Sudden Cardiac Arrest (SCA). Introducing Modified Early Warning Score (MEWS) charts (see Section 2) to patients' records allowed, despite its simplicity, to reduce the risk of SCA and hospital mortality [2]. In light of the above it is tempting to develop a real-time decision support system allowing automatic evaluation of the general condition of the patient, including suggestions of the most likely causes of the deterioration and advisable further procedures. It should be noted that automated real-time systems used so far are very simple and calculate only MEWS and its components; they do not give any additional information. None of such solutions have been used in Poland [3]. The discussed system may be especially important in places such as admission rooms and Accident and Emergency Departments.

The aim of the research presented in this work is to design such an advanced decision support solution, using as its inputs signals derived from a typical medical monitor. The system should, in particular, estimate the risk of Sudden Cardiac Arrest (SCA) in the monitored patients and include analysis of impedance cardiography and impedance pneumography curves.

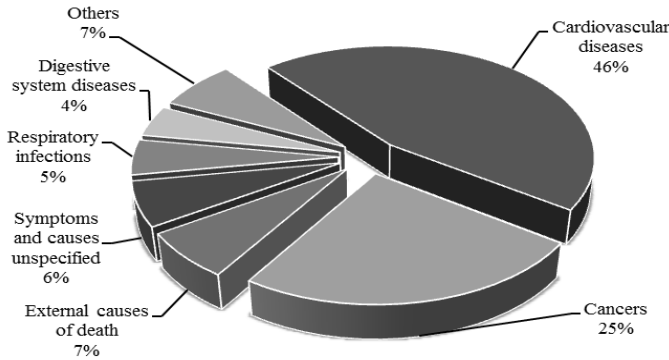


Fig. 1 The share of main causes of the total number of deaths in Poland, 2006 [1]

The remainder of this chapter is organised as follows: Section 2 presents the current approaches to the problem of quantitative estimation of the patient's health and its changes. Section 3 presents outline of the proposed system. Section 4 discusses the hardware and software considerations of the solution being developed. Closing thoughts are offered in Section 5.

2 Current Tools for Quantitative Estimation of Patient's Health

Difficulties in objective evaluation of how critical the patient's condition is, and the necessity to decrease the mortality in hospitals prompted doctors to introduce objective scales of that aspect of medical recognition. This section presents three examples of such solutions.

2.1 Modified Early Warning Score

Modified Early Warning Score (MEWS) is a simple index used to quickly determine the patient's condition. It is based on data derived from four physiological readings (systolic blood pressure, heart rate, respiratory rate, body temperature) and one observation – level of consciousness based on the Alert, Voice, Pain, Unresponsive (AVPU) scale. The results are compared to reference ranges, allowing generation of a single score through addition of scores resulting from the obtained readings – as presented in Table 1.

Table 1 A sample MEWS chart

Readings, observations	Score						
	3	2	1	0	1	2	3
Systolic blood pressure	<45%	30%	15% down	Normal for patient	15% up	30%	>45%
Heart rate [beats · min ⁻¹]	-	<40	41-50	51-100	101-110	111-129	>130
Respiratory rate [breaths · min ⁻¹]	-	<9	-	9-14	15-20	21-29	>30
Temperature [°C]	-	<35	-	35.0-38.4	-	>38.5	-
AVPU	-	-	-	A	V	P	U

The results of the conducted studies [2] suggest that the proportion of patients admitted and those who died in hospital increased significantly as the MEWS score increased ($p < 0.001$). Multivariate regression analysis further identified five independent predictors of hospital admission: systolic blood pressure ≤ 100 mm Hg, pulse rate ≥ 130 beats per minute, respiratory rate ≥ 30 breaths per minute, temperature $\geq 38.5^\circ\text{C}$ and an impaired level of consciousness.

MEWS index may encourage by its simplicity, confirmed effectiveness, especially in qualifications for intensive care unit, and the possibility of interpretation by the paramedics. However, it appears that the evaluation of the patient's general condition is much more complex process than simple summing ranks of several parameters.

2.2 Acute Physiology, Age, and Chronic Health Evaluation

Acute Physiology, Age, and Chronic Health Evaluation (APACHE III) is a score gathered using the method presented in [4] with 18 variables in acute physiological scores ranging from 0 to 252 (i.e. pulse, mean blood pressure, temperature respiratory rate, etc.), the age score from 0 to 24 and the chronic health evaluation from 0 to 23. The total scores in the APACHE III, summarized by the three mentioned categories, range from 0 to 299. The patients are classified according to etiology, major organ involved, surgical and medical status upon those categories. Patient's data like age, sex, race and preexisting comorbidities are registered prior to establishing his health status.

APACHE III scale allows for much more precise evaluation of patient's health than MEWS. However, applying this scale is cumbersome, as the number of input parameters is large. Although the score is calculated by a computer program, there are no automated solutions allowing to compute this score without the need to manually enter the parameters. This practically eliminates the possibility of using APACHE III as tool for continuous monitoring of the patient's state.

2.3 Therapeutic Intervention Scoring System

The Therapeutic Intervention Scoring System (TISS), introduced in 1974 [5], has become a widely accepted method of classifying critically ill patients. Since then the system has been updated, because of recent innovations in critical care. Unlike other scoring systems, TISS does not attempt to quantify a patient's severity of illness. Instead, it quantifies the intensity of care provided to an intensive care patient. It can be used to compute resource use and, to a lesser degree, nursing requirements. A variation of TISS system is TISS-28, used in Poland. Its input form consist of 28 yes/no questions, divided into 7 groups: basic actions, breathing, blood circulation, kidneys, metabolism, central nervous system and other interventions.

Similarly to APACHE III, use of TISS may be cumbersome due to a large number of input data that have to be entered manually. This is even more evident in the original version of TISS, which requires answers to 75 questions.

3 Overview of the System for Estimation of Patient's State

The idea of the proposed system is to evaluate the patient's condition and its changes using Artificial Intelligence (AI) techniques. In particular, a Bayesian Network (BN) is proposed as the way to encode the doctor's knowledge and provide means for reasoning under uncertainty – this will be further elaborated in Section 3.2. A key concept of the presented approach is to provide a way to automatically, without much aid from the hospital staff, supply the algorithm with necessary input data. To this end, the solution is being developed as an autonomous microprocessor system, attached to a typical medical monitor. Such set-up allows to gather all data produced by the monitor (including measurements: blood pressure, raw ECG curve, etc. and results of simple analyses, usually conducted by such monitors: e.g. parameters derived from the ECG curve).

The following paragraphs present, in order as they appear in the data processing path, the main functional blocks of the proposed solution.

3.1 Input communication Block

Interaction with the medical monitor is the primary task of the input communication block. Compatibility at hardware and transmission protocol level is crucial for such cooperation. It should be noted that the medical monitors transmits data continuously, which is especially useful to purposes of the discussed device, but also imposes certain requirements regarding processing speed of the device. More detailed description of the communication scheme is presented in Section 4.1.

Input communication block has also to provide a solution for interaction with medical personnel – the most important is possibility of entering information about the patient's consciousness. For this purpose an on-screen form taking advantage of the touch-screen capabilities is proposed.

3.2 *Signal Pre-processing Block*

Thanks to the use of medical monitor for collecting information about the patient, the need for data pre-processing is minimised. Nevertheless, some initial data analysis is necessary. Purpose of this analysis is, among others, to evaluate the usefulness of collected data and calculate some auxiliary indicators and parameters.

Among the above-mentioned parameters the most important factors are those regarding the ECG curve analysis. This curve is a rich source of data about the patient's health. In order to make best use of these data it is necessary to process the raw curve in order to provide some qualitative and quantitative indicators typically used in medical diagnosis. A medical monitor usually calculates some of them (e.g., ST segment level and elevation, detection of arrhythmias: premature supraventricular, ventricular, premature ventricular, bradycardia and tachycardia), but other have to be computed in the proposed device. A number of algorithms for calculation of parameters, such as for example T-wave alternans detection and deceleration capacity assessment in ECG signal, have already been designed and implemented by the team members; the reader is referred to [6] and [7] for detailed description of these solutions.

3.3 *Decision-Making Algorithms Block*

After signal pre-processing, described above, series of data records are available corresponding to all gathered information concerning the condition of the patient. On the basis of such set of data records, device should evaluate the condition of the patient, detect the risk of SCA, recognize features of cardiac electrical instability, acute heart failure, respiratory failure, dehydration or over-hydration, severe infection. Device should also specify suggestions for further proceedings. Proper operation of this module is crucial for functioning of the device – if the developed algorithms are able to capture threats without causing false alarms and suggest temporary procedure correctly, it will give substantial support in situations where immediate medical consultation is impossible (e.g., rescue teams, ambulances) and will likely reduce the risk of cardiac arrest during hospitalization period.

A number of different Artificial Intelligence (AI) approaches have been considered as the basis for this block, for example Neural Networks, Fuzzy Logic and Bayesian Networks. Of these, Bayesian Networks (BNs) [8] have been selected due to following advantages:

- ability to encode human knowledge about the domain – in this case doctor's knowledge about illnesses and their symptoms,
- ability to perform learning from data in order to adjust the human-created description or to create a completely new network – especially the first possibility is valuable, as it allows to self-adjust the algorithm based on observation of patients and doctor's diagnoses,

- ability to reason in case of missing data – which allows the algorithm to function even if some input data are not present, for example due to use of different medical monitor,
- proven usefulness in medical expert systems [9].

The authors also investigate the possibility of using a newer concept based on BNs, namely Multi-Entity Bayesian Networks (MEBNs) [10], which allow to combine BNs with first-order logic. This concept has not been yet applied in medical diagnosis.

It should be noted that the selection of AI algorithms impacts the choice of the hardware and software platform, described in Section 4. The proposed algorithms may be computationally intensive, therefore the hardware platform of the solution must provide significant computational power. Moreover, the available implementations of the algorithms may require specific software environment. This is especially true in case of MEBNs, as the only freely available mature implementation of this concept [11] is written in Java; for this reason availability of Java Virtual Machine was a crucial factor in the choice of the operating system, as described in Section 4.3.

3.4 Output Presentation and Data Storage Block

The main goal of this block is to present the analysis result in a clear and easy to interpret form. For this purpose a solution using an LCD screen is proposed. The system should also be capable of providing audible indication of an emergency condition. Another task of this block is to store the results of analyses. A popular format of data storage, enabling easy downloading of the saved results to a Personal Computer should be employed – examples may include USB mass storage and Secure Digital (SD) card as the hardware platform or text and Microsoft Excel (xls,xlsx) files as the software format.

4 Hardware and Software Considerations

This section presents the discussion that led to the choice of particular hardware and software components.

4.1 Medical Monitor

As the system itself does not have any sensors, it has to interact with a medical monitor. For this reason, a monitor capable of continuous transmission of measurement results, preferably utilising a popular transmission medium and format, is required. The choice of the monitor used during the development phase is important, as it will impact the design of the system in both hardware and software layer.

The analysis of typical solutions present in commercially available monitors shows that most of today's monitors are capable of outputting the data (measurements and analyses) in a computer-friendly form. The approaches can be roughly divided into two categories:

- continuous transmission – typically using 10BASE-T or 100BASE-TX Ethernet (IEEE 802.3i, IEEE 802.3u) and TCP/IP stack,
- file storage – typically using USB mass storage device or Secure Digital memory card and common file types (e.g., Microsoft Excel).

Of these, only the first approach can be used in the proposed solution. However, unlike the file storage, this approach uses proprietary data format. In order to successfully transfer data from the monitor to the discussed system the format has to be known. The best approach here is to work closely with the manufacturer of the monitor in order to obtain details of the protocol; although reverse-engineering of the protocol is also possible, it should be avoided in the problems relating to human health as possibly unreliable.

Considering the discussed requirements an Emtel FX2000-MD medical monitor (Figure 2) has been selected as the data source for the discussed system. It is capable of transmitting the data in real time over an Ethernet - UDP connection and the manufacturer made available – for the research purposes – the details of the data format.

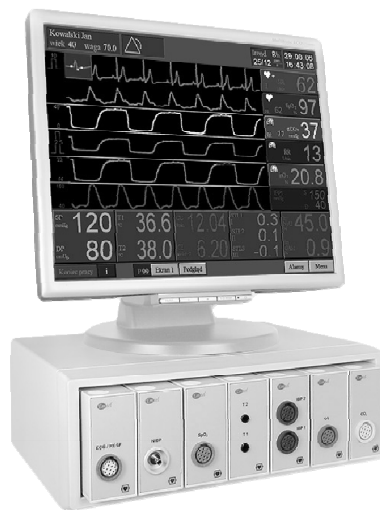


Fig. 2 Emtel FX2000-MD medical monitor [12]

Based on the information from the manufacturer a PC program for receiving data from the medical monitor has been written. The aim of writing a PC program (instead of a program for the target hardware) was to provide an easy to debug solution for sorting out all potential problems concerning the transmission; another reason was to obtain data for the early stage of the algorithms development. The program proved that the transmission is stable and the percentage of datagrams lost is negligible.

4.2 Hardware of the Discussed System

Section 3 outlined some requirements for the hardware of the proposed system. These included: significant computing power, comfortable-sized LCD touch-screen, ability to produce audible alerts, ability to connect to the medical monitor (which translates to an Ethernet interface, as explained in Section 4.1), USB and SD interfaces. Moreover, the hardware of the system should be characterised by compact dimensions and sturdy construction. Mains operation is mandatory, optional battery operation is a plus.

The most straightforward choice fulfilling these requirements is a notebook computer equipped with a touch-screen – such notebooks have appeared recently in the market. However, this choice has its drawbacks. The most important is the form factor of such a unit: two elements connected by hinges and an external power supply do not guarantee required sturdiness, especially if the device is to be used in ambulances. Moreover, a notebook may not be able to produce audible alerts of required loudness, and adding external speaker would further aggravate reliability problems.

Another possible solution would be a tablet computer, or a large-screen smartphone. However, here the problem of several components wired together (main unit, power supply, speaker, possibly a dongle/converter for Ethernet interface) is also present.

It should also be noted that using standard computer equipment may lead to unwanted user behaviour (for example – installing additional software that could interfere with the main application). It has also been noted (in personal communication with doctors) that the use of such equipment in a life-saving environment may be seen as unprofessional.

Due to the considerations explained above it has been decided to build a custom device, based on an embedded microprocessor with considerable computational power. Another advantage of such solution is the possibility to employ specialised Digital Signal Processors (DSP), which could be utilised in implementation of the AI algorithms. Because of previous experience of the team, Texas Instruments DaVinci DM3730 Digital Media Processor [13] has been chosen, which integrates in one integrated circuit an ARM Cortex-A8 processor, a TMS320C64x+ DSP and a POWERVR SGX graphics accelerator. In order to speed-up the prototyping process, a commercially available board equipped with this processor has been chosen as the prototyping platform: Embest DevKit8500 (Figure 3).

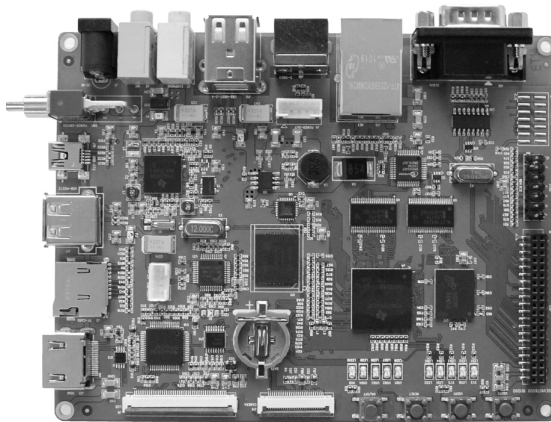


Fig. 3 Embest DevKit8500D Evaluation Board [14]

4.3 Operating System

After choosing the hardware platform, the next step is to select the operating system. The Embest DevKit8500 prototyping board is supplied with a number of operating systems: Android, Ångström (embedded Linux distribution), Windows CE and Linux (kernel only). Out of these, the Linux kernel together has been chosen and the Ubuntu distribution has been added to it due to the following considerations:

- availability of a Software Development Kit provided by Texas Instruments for DaVinci 3730,
- good support for Java Virtual Machine, which is needed for at least one algorithm implementation (see Section 3.3),
- good support for peripherals, which can be added at a later stage of the project.

It should be noted that creating a functioning operating system required a number of steps, including configuration and compilation of Linux kernels, modules, additional drivers, filesystem hierarchy (rootfs) of the Ubuntu distribution, creating star-up SD disk and adding required applications – in particular the Java virtual machine. In particular, the configuration of kernel has not been straightforward, as the DaVinci processor differs significantly from the typical x86/x64 microprocessor.

The works resulted in fully-functioning Linux OS based on the Ubuntu distribution. It is a full-fledged platform for the development of the discussed system, allowing running GUI programs and also making possible to use features of the embedded DSP. The only problem still present is the memory limit, which currently is set at 256 MB due to complexity of assigning separate memory blocks to the ARM core and the DSP.

5 Conclusions and Further Work

This chapter presented a discussion of the approach chosen for development of the system for the estimation of patient's health. The concept of the system has been presented, which allowed to enumerate the requirements for the hardware and software platform. Considering these requirements, a discussion of possible approaches has been presented and the choice justified. Works leading to the creation of a fully-functional operating system for the selected microprocessor board have been described, as well as the successful implementation of the communication protocol for obtaining data from the medical monitor.

Further work will focus on software implementing the functionality described in Section 3, especially the AI algorithm for estimation of the patient's health.

Acknowledgements. This project has been financed through National Science Centre funding granted by decision number DEC-2011/01/B/ST6/04726.

References

- [1] Wojtyniak, B., Goryński, P., et al.: Sytuacja zdrowotna ludności Polski. Narodowy Instytut Zdrowia Publicznego – Państwowy Zakład Higieny, Warszawa (2008)
- [2] Burch, V.C., Tarr, G., Morroni, C.: Modified early warning score predicts the need for hospital admission and in-hospital mortality. *Emerg. Med. J.* 25, 674–678 (2008), doi:10.1136/emj.2007.057661
- [3] Tarassenko, L., Hann, A., Young, D.: Integrated monitoring and analysis for early warning of patient deterioration. *British Journal of Anaesthesia* 97(1), 64–68 (2006)
- [4] Knaus, W.A., Wagner, D.P., Draper, E.A., Zimmerman, J.E., Bergner, M., Bastos, P.G., Sirio, C.A., Murphy, D.J., Lotring, T., Damiano, A., et al.: The APACHE III prognostic system. Risk prediction of hospital mortality for critically ill hospitalized adults. *Chest* 100(6), 1619–1636 (1991), doi:10.1378/chest.100.6.1619
- [5] Miranda, D.R., de Rijka, A., Schaufeli, W., et al.: Simplified Therapeutic Intervention Scoring System: The TISS-28 items. Results from a multicenter study. *Crit. Care Med.* 24, 643 (1996)
- [6] Kamiński, M., Chłapiński, J., Sakowicz, B., Kotas, R., Napieralski, A.: ECG Signal Processing for Deceleration Capacity Assessment. *Elektronika - Konstrukcje, Technologie, Zastosowania* (December 2011) ISSN 0033-2089
- [7] Kamiński, M., Chłapiński, J., Sakowicz, B., Kotas, R.: ECG Signal Processing for T-wave Alternans Detection. In: 16th International Conference Mixed Design of Integrated Circuits and System, Łódź, pp. 623–628 (2009) ISBN 978-839-2875-60-4
- [8] Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco (1988) ISBN 978-155-8604-79-7
- [9] Pourret, O., Naim, P., Marcot, B.: *Bayesian Networks: A Practical Guide to Applications*. John Wiley and Sons (2008) ISBN 978-047-0060-30-8
- [10] Laskey, K.B.: MEBN: A Language for First-Order Bayesian Knowledge Bases. *Artificial Intelligence* 172(2-3), 140–178 (2008)

- [11] Carvalho, R.N., Laskey, K.B., Costa, P.C.G., Ladeira, M., Santos, L.L., Matsumoto, S.: *UnBBayes: Modeling Uncertainty for Plausible Reasoning in the Semantic Web*. IntechWeb, Croatia (2010) ISBN 978-953-7619-54-1
- [12] Emtel, FX2000-MD medical monitor manufacturer homepage (2012), <http://www.emtel.com.pl> (accessed October 10, 2012)
- [13] Texas Instruments, DM3730 Digital Media Processor description (2013), <http://www.ti.com/product/dm3730> (accessed January 15, 2013)
- [14] Embest Technology, DevKit8500D Evaluation Board description (2013), <http://www.armkits.com/product/devkit8500d.asp> (accessed January 15, 2013)

Dependability Aspects of Autonomic Cooperative Computing Systems

Michał Wódczak

Department of Information Technology
Poznań University of Economics
ul. Mansfelda 4, 60-854 Poznań, Poland
e-mail: mwodczak@kti.ue.poznan.pl

Abstract. As the number of globally interconnected devices has become substantial, the resulting systems are getting prone to configuration issues and dependability becomes one of their key characteristics. Following the rationale behind self-management in autonomic computing, the main trend is to put emphasis on the ability of a computer system to self-configure, self-optimize, self-heal, and self-protect without any explicit need for external human intervention. This is crucial for complexity reasons, as a complete automation appears to be the only reasonable and justified way forward. This paper advocates for the possibility of adjusting the level of dependability with the aid of Autonomic Cooperative Behaviour, expressed through cooperative data processing and computing. In particular, devices may improve the related system robustness by sharing their computational resources for the purposes of cooperative data transmission. Autonomic system design, in turn, is expected to guarantee system stability and scalability, especially in the case of very large set-ups composed of a significant number of devices exposing such Autonomic Cooperative Behaviour simultaneously. For this reason, a properly adjusted overlay autonomic network architecture needs to be employed so the overall system may be controlled by specific Decision Making Entities interacting among themselves and operating with the aid of control loops.

1 Introduction

Since the number of devices interconnected worldwide is growing drastically, the question of dependability of ubiquitous computing systems has become substantial [1]. In fact, dependability translates into the provision of the following key features: reliability, availability, safety, confidentiality, integrity, and maintainability [2]. What is more, it may be further integrated with autonomic computing advocating for self-configuration, self-optimization, self-healing, and self-protection [3]. This paper discusses a new approach, not only capitalising on the above-mentioned concepts but also incorporating the novel notion of Autonomic Cooperative Behaviour (ACB). The idea stems from the rationale behind distributed

cooperative system, aiming to enable collaboration among devices. This way the devices are expected to share their computational capabilities and memory to perform joint data processing for the benefit of meeting the global performance indicators through increased dependability. Obviously, as such a system grows, it appears necessary, from the local perspective, to instantiate cross-layer integration of certain aspects with the use of a network protocol in order to limit any unnecessary control overhead resulting from the exchange of data among the cooperating devices. However, today's distributed systems are becoming more and more complex. For this reason, the transition from a local to the global context additionally requires the incorporation of an autonomic overlay so the distributed cooperative system may self-manage. On the one hand, devices may improve system dependability by sharing their computational resources by expressing Autonomic Cooperative Behaviour through cooperative data processing and transmission. On the other hand, the paradigm of autonomic system design assumes that such a computing system should follow the operating principles of an autonomic nervous system and, thus, be able to self-configure and, then, self-manage without any external human intervention [4]. In fact, both cooperation and autonomics may be welded into a joint concept to offer best service possible to end users under the assumption of strictly following the externally imposed policies [5]. Consequently, such a system would have to express the ability to analyse the current situation and make an attempt to benefit from the gains offered by cooperative processing and computing. The paper is organised as follows. First, the concept of Cooperative Computing and Processing is outlined in Section 2. It is complemented in Section 3 where the orchestration of Autonomic Cooperative Behaviour is described through the incorporation of network layer protocol. Following, the issue of Autonomic System Design is discussed in Section 4 from the global perspective. The paper is concluded in Section 5.

2 Cooperative Computing and Processing

In general, a set of devices attempting to obtain a certain common goal might need to express Autonomic Cooperative Behaviour by performing, for example, cooperative computing and processing of the code words necessary for the establishment of the communication between the Source Node (SN) and Destination Node (DN) [6]. This way, such cooperating nodes are sharing their computational power and memory necessary for the completion of this task. This approach may be crucial in the case of battery-powered systems where distributed code word computation and processing obviously means that each device is able to preserve energy and function longer, compared to the case of a typical single node processing [7]. The notion of such Autonomic Cooperative Behaviour among devices may be illustrated as depicted in Fig. 1.

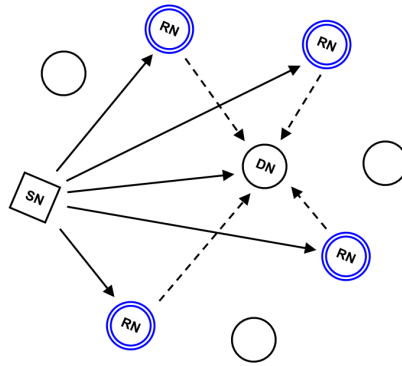


Fig. 1 Autonomic Cooperative Behaviour among computing nodes

In such a case, a group of Relay Nodes (RN) assists in the transmission between SN and DN by performing autonomic cooperative computation of the code words in a synchronous manner. In particular, the cooperation schemes leading to the generation of specific spatio-temporal code words may be generally described with the use of the example cooperation matrices (1), (2), and (3) [8], [9].

$$G_2 = \begin{bmatrix} x_1 & x_2 \\ -x_2^* & x_1^* \end{bmatrix} \tag{1}$$

$$G_3 = \begin{bmatrix} x_1 & x_2 & x_3 \\ -x_2 & x_1 & -x_4 \\ -x_3 & x_4 & x_1 \\ -x_4 & -x_3 & x_2 \\ x_1^* & x_2^* & x_3^* \\ -x_2^* & x_1^* & -x_4^* \\ -x_3^* & x_4^* & x_1^* \\ -x_4^* & -x_3^* & x_2^* \end{bmatrix} \tag{2}$$

$$G_4 = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ -x_2 & x_1 & -x_4 & x_3 \\ -x_3 & x_4 & x_1 & -x_2 \\ -x_4 & -x_3 & x_2 & x_1 \\ x_1^* & x_2^* & x_3^* & x_4^* \\ -x_2^* & x_1^* & -x_4^* & x_3^* \\ -x_3^* & x_4^* & x_1^* & -x_2^* \\ -x_4^* & -x_3^* & x_2^* & x_1^* \end{bmatrix} \tag{3}$$

The number of columns contained in each matrix corresponds to the number of RNs, while the number of rows denotes the consecutive time slots. The use of such distributed codes is advantageous because the system robustness and, therefore, dependability increases, thanks to the diversity gain, as depicted in Fig. 2. The diversity gain results from a specific computation of the distributed spatio-temporal code word matrix so the orthogonality criterion is satisfied [8].

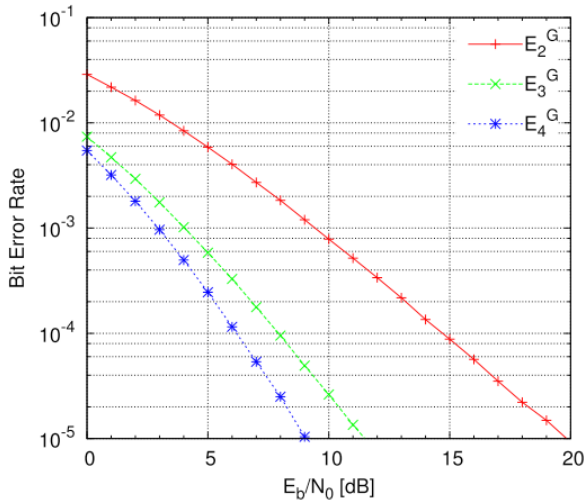


Fig. 2 Increased dependability of transmission thanks to cooperative computing and processing

3 Orchestration of Cooperative Behaviour

The approach presented in Section 2 is well tailored to a small group of devices, expressing Autonomous Cooperative Behaviour on a local scope. However, usually there are more devices available and making a proper selection of the nodes intending to expose such Autonomous Cooperative Behaviour might be very advantageous in terms of conserving energy [10].

For this reason one might apply the mechanisms of the Optimised Link State Routing (OLSR) protocol devised for mobile ad-hoc networks [11]. The protocol belongs to a proactive class, which means that the control messages are disseminated on a periodical basis. Consequently, the information regarding the status of distinct links is available almost without any delay.

In particular, OLSR is equipped with the Multipoint Relay (MPR) selection heuristics aiming to optimise the protocol overhead during the phase of topology recognition [12]. This heuristics is performed with the use of both the one-hop and two-hop neighbour sets identified with the use of Hello messages. More specifically, each node n belonging to the one-hop neighbourhood $N(x)$ of a given source

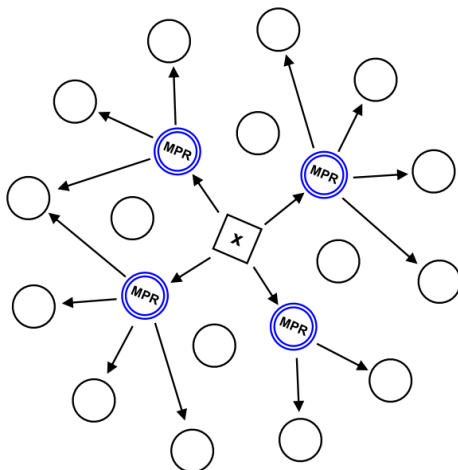


Fig. 3 Multi-Point Relay station selection heuristics

node x disseminates Hello messages. Such messages contain the addresses of all the one-hop neighbours discovered by the node n , as well as the corresponding link codes [11]. Based on data of this type, the node x is able to acquire the relevant knowledge about its two-hop neighbourhood $N^2(x)$ reachable through node n . What is more, having identified its entire one-hop and two-hop neighbourhoods, node x can select its MPRs according to the following heuristics, outlined below [11].

First, node x includes in its $MPR(x)$ set all the symmetric one-hop neighbouring devices being the only ones to provide reachability to a node n^2 located in the strict symmetric two-hop neighbourhood and additionally are always willing to carry and forward traffic. Next, until there still exist any uncovered nodes in $N^2(x)$, the heuristics keeps on selecting such a node belonging to $N(x)$ which has not been inserted into the $MPR(x)$ set yet and is characterised by the highest willingness to carry and forward traffic. In the case of a multiple choice, the one is chosen which provides the highest reachability $R(n)$, i.e. through which the highest number of still uncovered nodes in $N^2(x)$ may be reached. Otherwise, if it is impossible to select a single device, the one of the highest degree is chosen, where the degree $D(n)$ of a one-hop neighbour denotes the number of its symmetric neighbours, excluding all the members of $N(x)$ and the node x performing the computation.

Looking at both Fig. 1 and Fig. 3 one may notice that such a goal may be obtained with the use of the fact that the groups exposing Autonomic Cooperative Behaviour can be formed on the basis of the assignment of devices to MPR set(s) [10]. In other words, first, each neighbour n having zero degree $d(n)$ needs to be removed by node x from the set $N(x)$. Then the classic MPR selection heuristic is performed iteratively, over the set $N(x)$, until all the nodes it contains have been pre-selected into distinct cooperative sets $C(x, n^2)$ and, additionally, assigned to the relevant redundant MPR sets. In other words, with every iteration, a new set

$MPR^i(x)$ is created and each of the nodes it contains is also included in the proper cooperative set $C(x, n^2)$. This set will provide the capability of Autonomous Cooperative Behaviour through cooperative computational processing and transmission of the data received from the source node x towards the destination node n^2 , where n^2 belongs to the two-hop symmetric neighbourhood of x . It also means that any intermediate node n can be included in more than one set expressing Autonomous Cooperative Behaviour [10].

4 Autonomous System Design

Autonomic computing was introduced as a means of mapping the rules governing the autonomic nervous system onto computer systems [4]. Soon after, this idea was extended to embrace networked set-ups through the incorporation of autonomic cooperative networking [5]. This section discusses the further evolution of the relevant architectural aspects of this concept, related to the notion of Autonomous Cooperative Behaviour, into autonomic cooperative computing, as a fusion of both the approaches. In particular, the devices forming an autonomic cooperative network are expected to expose Autonomous Cooperative Behaviour (ACB) instantiated, in this case, by the notion of cooperative computing necessary for the aforementioned spatio-temporal code word computation. Consequently, such devices may share their computational power and memory to perform cooperative processing, as instructed by the overlay Decision-Making Entities, being the inherent components of the currently being standardised Generic Autonomic Network Architecture (GANA) [13].

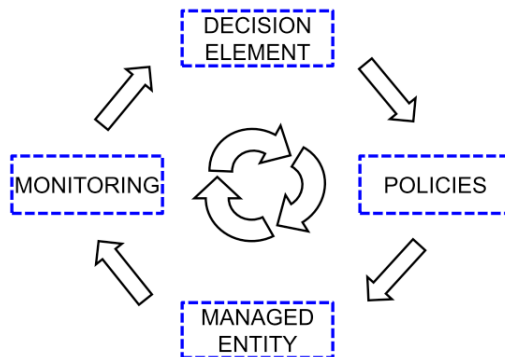


Fig. 4 Autonomic control loop

As it has been already signalled, GANA architecture assumes the existence of control loops and their respective Decision Elements (Des) at the network, node, function, and protocol level [14]. This way each DE has its exclusive responsibilities and, at the same time, it is able to interact with other DEs. It means that, based on the control and monitoring information available within a control loop, each

DE is able to take its autonomic decisions which might be to some extent affected by the data it exchanges with its higher/lower level, peering or sibling counterparts. Consequently, even a minor event may be the reason for triggering certain Autonomic Cooperative Behaviour that might involve a number of nodes or even the whole system for reasons such as the overall dependability [15].

Particularly, autonomic systems are especially assumed to be able to self-discover which may pertain to many aspects at the same time, including e.g. service discovery, topology discovery, fault discovery, etc. In fact, this can further enhance the effectiveness of autonomic configurations tailored for optimum cooperation in terms of dependability. In order to avoid conflicting decisions that would negatively affect the system performance and, thus, dependability, the hierarchical interactions among DEs need to be employed to guarantee system stability and scalability. Consequently, not only should the architecture enable specific devices to express Autonomic Cooperative Behaviour, but also the system should be autonomic as a whole [5]. As it was already mentioned, an autonomic system must continuously monitor itself and be able to align its current operation with the requirements arising either from monitoring data or simply imposed by policies that may be changing over time. As a result, in certain circumstances, the directions of a higher-level DE may limit the freedom of a given DE to make decisions based on the data available locally. The decisions might need to be taken at the system level in order to make it feasible for the overall system to perform tasks of a global scope. At the same time, different operations, such as Autonomic Cooperative Behaviour between or among devices can be carried out without any interruption, as long as this does not result in any violation of the rules.

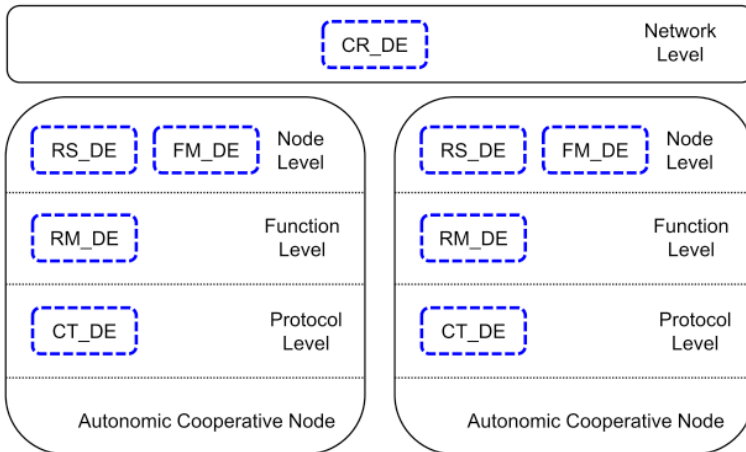


Fig. 5 Autonomic Cooperative Behaviour from architectural perspective

To accommodate such functionality, it is primarily assumed that the definition of the Autonomic Node is enhanced with the notion of Autonomic Cooperative Behaviour, which results directly in the introduction of an Autonomic Cooperative Node as an enabler for cooperation among the Managed Entities, orchestrated by their corresponding Decision Elements (Figure 6). In particular, GANA defines the aforementioned four levels on which decision entities may appear [5]. Starting from the protocol level, there is a new Cooperative Transmission Decision Element (CT_DE) introduced which is responsible for controlling the aspects of cooperative transmission protocol requiring the cooperative computation of code words. The operation of CT_DE has to be aligned with the already existing Routing Management Decision Element RM_DE, located on the function level, which needs to interact with its sibling DEs, so the computation of the routing tables maintained at the cooperating devices is properly synchronised [10]. What is more, the RM_DE also needs to act pursuant to the directions from the other existing DEs, i.e. the Resilience and Survivability Decision Element RS_DE and the Fault Management Decision Element FM_DE, both located on the node level and responsible for dependability [5]. Specifically, the RS_DE is assumed to cover aspects of service resilience and survivability. At the same time, it is supposed to interact with FM_DE which, in turn, controls the symptoms suggesting that a failure, for example in terms of service continuity, may be imminent. Finally, while all these DEs are located within Autonomic Cooperative Nodes, it is still necessary to provide substantial coordination on the network level. This task is accomplished with the aid of the Cooperative Re-Routing Decision Element CR_DE, which is responsible for overseeing the system situation from a higher-level perspective [5].

5 Conclusion

In this paper the dependability aspects of autonomic cooperative computing systems were presented from the cross-layer and architectural design perspective. In particular, the notion of Autonomic Cooperative Behaviour was explained which may be instantiated by collaborative computing to help increase the level of resilience and, therefore, also dependability. Then the broader context of the network layer was analysed where the MPR selection heuristics of the OLSR protocol was employed as a means of the orchestration of the above-mentioned cooperative processing in terms of the spatio-temporal distributed code word computation for increased data transmission robustness. Finally, the autonomies of the overall system was taken into consideration and the whole solution was put under the umbrella of the GANA architecture. This way one may expect the system operate without any explicit need for external human intervention, where Autonomic Cooperative Behaviour among network nodes is employed as decided by the system itself.

Acknowledgement. This paper summarises the most recent work performed by the author within ETSI ISG AFI (Industry Specification Group on Autonomic network engineering for the self-managing Future Internet) and is a follow-up to the previous research for EU FP7 projects EFIPSANS and E-SPONDER at Telcordia Technologies (Ericsson).

References

- [1] Knight, J.C.: An introduction to computing system dependability. In: 26th International Conference on Software Engineering, ICSE 2004 (May 2004)
- [2] Lu, Y.: Dependability Aspects of Ubiquitous Computing. In: The Second International Conference on Availability, Reliability and Security, ARES 2007 (April 2007)
- [3] Sterritt, R., Bustard, D.: Autonomic Computing - a means of achieving dependability? In: 10th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (April 2003)
- [4] Kephart, J.O., Chess, D.M.: The vision of autonomic computing. *IEEE Computer* 36(1) (2003)
- [5] Wódczak, M.: *Autonomic Cooperative Networking*. Springer, New York (2012)
- [6] Doppler, K., Redana, S., Wódczak, M., Rost, P., Wichman, R.: Dynamic resource assignment and cooperative relaying in cellular networks: Concept and performance assessment. *EURASIP Journal on Wireless Communications and Networking* (July 2009)
- [7] Wódczak, M.: *Autonomic Cooperative Networking for Wireless Green sensor Systems*. *International Journal of Sensor Networks (IJSNet)* 10(1/2) (2011)
- [8] Alamouti, S.: A Simple Transmit Diversity Technique for Wireless Communications. *IEEE Journal on Selected Areas in Communications* 16(8), 1451–1458 (1998)
- [9] Tarokh, V., Seshadri, N., Calderbank, A.R.: Space-Time Codes for High Data Rate Wireless Communication: Performance Criterion and Code Construction. *IEEE Transactions on Information Theory* 44(2), 744–765 (1998)
- [10] Wódczak, M.: Extended REACT – Routing information Enhanced Algorithm for Cooperative Transmission. *IST Mobile and Wireless Communications Summit* (June 2007)
- [11] Clausen, T., Jacquet, P.: Optimized Link State Routing Protocol (OLSR). RFC 3626 (October 2003)
- [12] Qayyum, A., Viennot, L., Laouiti, A.: Multipoint relaying for flooding broadcast messages in mobile wireless networks. In: *HICSS 2002* (January 2002)
- [13] Wódczak, M., Meriem, T.B., Chaparadza, R., Quinn, K., Lee, B., Ciavaglia, L., Tsagaris, K., Szott, S., Zafeiropoulos, A., Radier, B., Kielthy, J., Liakopoulos, A., Kousaridas, A., Duault, M.: Standardising a Reference Model and Autonomic Network Architectures for the Self-managing Future Internet. *IEEE Network Magazine* 25(6) (November 2011)
- [14] Chaparadza, R., Papavassiliou, S., Kastrinogiannis, T., Vigoureux, M., Dotaro, E., Davy, A., Quinn, K., Wódczak, M., Toth, A.: Creating a viable Evolution Path towards Self-Managing Future Internet via a Standardizable Reference Model for Autonomic Network Engineering. In: *Future Internet Conference*, Prague (2009)
- [15] Wódczak, M.: *Future Autonomic Cooperative Networks*. In: *Second International ICST Conference on Mobile Networks & Management 2010*, Santander, Spain (September 2010)

Life Cycle Cost through Reliability

Manac'h Yann-Guirec, Benfriha Khaled, and Aoussat Améziane

Arts et Métiers ParisTech - LCPI,
151 Blvd de l'Hôpital 75013 Paris, France
{yann-guirec.manach,khaled.benfriha,
ameziane.aoussat}@ensam.eu

Abstract. This work presents a novel approach using dependability of sub-assemblies to compute the life cycle cost of complex products (i.e. products that are technologically more complex than average or have a higher life expectancy) This work is composed of two parts, first a retrospective on life cycle cost and its challenges, then a description of the approach using reliability as the key element. This approach combines the usage of already well-known components or sub-assemblies alongside new, innovative ones in order to compute the life cycle cost of the system. This work is currently conducted as a PhD thesis and as industrial support.

1 Introduction

Many products and systems are more complex than others. That is mainly due to their technological complexity or life expectancy. These products therefore should have control over their availability, with maintainability taken into account. The use life phase is to generate huge costs in terms of maintenance, operation and energy consumption [1-2].

These products require a focused study of their design; this work presents a retrospective of life cycle cost analysis as well as the interactions of life cycle cost with the design phase. In a second part, we propose a novel approach based on life cycle cost estimation using reliability as a main factor. This approach allows the incorporation of innovative sub-assembly into a well-known system and focuses on the modification of the life cycle cost induced using the predicted reliability of the component. This work is currently conducted as a PhD thesis, with the support of industrial partners.

2 Life Cycle Cost Analysis

The first objective of life cycle cost analysis is to assess the monetary weight of each and every of the phases of the life of a product [3]. One of the many definitions of a product life phases is by Kriwet, Zussman & Seliger [4] on figure 1.

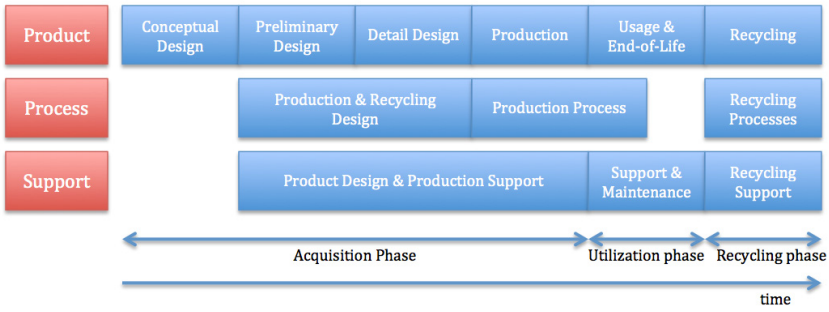


Fig. 1 Life Cycle Phases (adapted from [4])

The metaphases of the life of the product are well defined (Acquisition, Utilization & Recycling). Each of these phases are then sub-divided into multiple sub-phases than covers the whole life of the product. This definition is interesting as it identifies not only the product itself but also the process and logistic support needed to the accomplishment of the product-centric phases.

2.1 Key Phase

In a global life cycle cost approach, the key phase is the early conception phase. This phase is the key point where most of the technical choices are committed, these choices are of the same importance whether they are technological, components or materials. All these choices have a major impact on the detailed design phase. Once committed, they are not to be questioned due to cost of changing already committed choices.

The preliminary design phase is the one where most of the costs are committed (see figure 2). Thereby this phase has a disproportionate influence on the

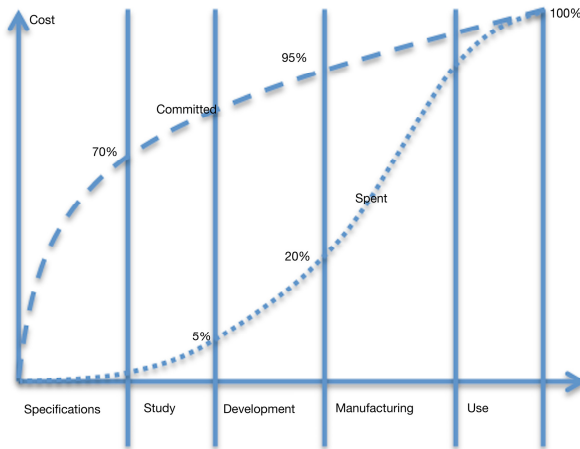


Fig. 2 Cost Phases (adapted from [])

downstream design, usage and end-of-life phases. At this stage, not all the necessary data for the assessment of the life cycle cost. That is even more evident in the case of a project containing many breakthrough innovations on which there is not yet any capitalization in terms of life cycle cost.

2.2 *Over-Costing and Under-Costing*

As previously shown, costs are incurred at a very early stage before being spent as the result of the project. There is therefore several risks that can be discerned for the rest of the life of the project.

The foremost influent factors in the case of complex products are, between others, inflation, financial costs and technological risks. This risks lead to two meta-risks: conservative calculation (over-costing) and liberal estimations (under-costing).

Over-costing can jeopardize the first phases of go/no-go without representing the true costs, therefore threatening commercial projections (or tender response). This is most often the case when there is over-estimation of costs from energy, personal, formation and/or maintenance.

Under-costing has the same effect, in a somewhat different view. Under-costing has a tendency to appear on projects where the risk involved with the industrialization phase is not well assessed and create huge costs, for instance when specific tooling is necessary but was not thought of.

3 Evolution of Costing

Life cycle cost analysis started to be used in the late 60s, early 70s by the US Department of Defense (US DoD) [5]. The DoD started to completely reorganize its procurement strategies using the deterministic factor of global costs instead of just using acquisition costs. Global costs included, but not limited to, support, fabrication, technological developments and formation costs. This started a *Design-to-X* [6] procedures that are used to minimize life cycle costs for system development and usage.

3.1 *A Transition from Programs to Projects*

Life cycle cost analysis slowly evolved from the DoD and large-scale procurement programs toward other US Army department at smaller scales. The DoD authored many notes concerning the *Design-to-Cost* processes and *Integrated Logistic Support* [7-8-9].

Integrated Logistic Support (ILS), also contributed by the US Army, ensures the supportability (maintenance, repair & operations) is taken into account within the design & development of a product/equipment.

This methodology has a major influence on design processes, especially that there is a tendency to search and identify as soon as possible reliability problems. This trend tends to initiate a dialogue on the drawing of parts thought for reliability improvement, maintainability, testability and/or availability [10-11-12-13].

ILS also has a huge part for the support in staff training, documentation and spare parts supply. All these considerations allow, and facilitate, specification steps, design steps and support phases, with a major focus through the whole process on maintainability.

The transition between calculating a global supply cost and seeking the minimal life cycle cost using maintenance also helped the usage transition of these methods from multiple large-scale systems to smaller scale products. Notes authored by the US Army [11-12-13] were published in the mid-80s; and, by late 80s, these concepts started to diffuse into civil design [14] first to avionic industries, electrical power production, oil & gas and railway industries [15-16-17-18].

This techniques diffusion coincides with a modification into the ILS approach; global life cycle (cf. figure 1) is more and more used for design, alongside with maintainability in complex and/or large-scale systems.

3.2 The Concept

Integration of these techniques in more and more industries also allows the refinement of the different methodologies that are used.

In fact, it is difficult whenever there is innovation into a project to estimate a life cycle cost (regardless of the complexity). This difficulty mostly arise from the fact that it is not possible to have concrete results that fits into the global life cycle cost model of the product from testing.

3.2.1 Innovative Difficulty

This difficulty, inherent to the conception of the innovative system, can jeopardize a life cycle cost model analysis conducted too early in the project. Also, difficulty arising from downstream design phase, and/or production, can lead to review the first estimators of life cycle cost.

The innovative part can cause huge repercussions on life cycle cost without changing the product a lot. One example of that is automotive paint with the transition from solvent based to oil based [19]. The two techniques have very small difference in costs regarding research, manufacturing and deposit onto the vehicle. However, aging is totally different, thus creating a modification on the usage phase of life cycle cost without changing upstream phases; this is also true for disposal procedures that are way heavier for solvent based paints.

The difficulties introduced by innovation can therefore deeply intervene into one phase of the life cycle cost, and have little or no impact on the other phases. However, it is extremely difficult to quantify this impact without having experts and trusting their judgment while waiting for field-tests that will confirm (or deny) their projections.

3.2.2 Estimators

The problem of estimating life cycle cost is not only present with innovations. It is also extremely difficult to assess during the evolution of a (well) known product in order to make the necessary adjustments to life cycle cost projections.

Several life cycle cost estimation techniques exist, however they all have advantages and disadvantages. These techniques are also linked to the fact that most of the design methods used in Western countries put forward the cost factors [20].

The three main types of life cycle cost estimators are:

- Analogue models from a similar system, extended over the current product using experts' interventions to model the transformation of the life cycle cost between the old and new products;
- Analytical models involving a set of experts modeling each and every component and/or sub-assemblies and then generating an overall life cycle cost model;
- Predictive analysis based on field-collected statistical data and/or probabilities related to the various components.

The first two types of estimators have the huge drawback of relying on the expertise of a small number of people and on the previously collected data [21]. These methods are generally deployed late in the design process, i.e. when the product is already undergoing production and not in an upstream design phase.

As shown by Baguley & al [22], the biggest challenge in cost modeling methods is identifying and collecting the necessary data for the construction of the model. At an upstream stage of the design process, there is little to no available data and information to identify the correct model.

In addition, another problem arises from highly innovative products. These products suffer from a lack of historical data linked to the innovative attributes; therefore it cannot be extrapolated into an already existing life cycle cost model [23]. This lack of data requires projection tools that can rely on data other than those usually harvested during the use phase.

An interesting development is to use a behavioral model of the product life cycle modeling (in terms of cost) and its failure. From this model, it is possible to create alternative models of the life cycle cost in terms of maintenance, and then go back to the overall life cycle cost.

4 Reliability Approach

The use of reliability to calculate life cycle cost is possible at an early design stage. This approach is based on the fact that each component or sub-assembly of a product as a reliability (function of charge, desired dependability, expected lifetime) that can be associated with it. Most often than not, most of the components used in a product are well described and only a handful are real innovations.

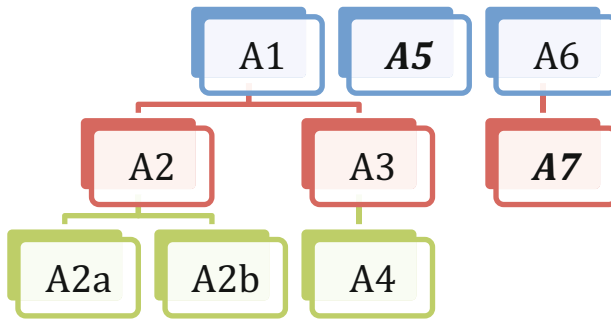


Fig. 3 Product functional nesting diagram

Figure 3 presents a product that uses nested sub-assemblies and top assembly, where italic represents innovations (A5 innovative assembly & A7 innovative sub-assembly). Each assembly and sub-assembly has its specific costs for engineering, manufacturing, support and disposal. Non-innovative sub-assemblies are perfectly defined using the collected data on already existing products. A7, the innovative sub-assembly is also likely to change the compartment of A6 its top assembly.

It is then possible, using simulation and expected data from design, to calculate the associated costs of the innovative assemblies. It is also necessary to introduce uncertainty into the calculation for the innovative components; hence not having a straight curve output but a region for life cycle cost estimation (see figure 4).

Another uncertainty source is the addition of the different project component life cycle cost, as discussed in 3.2.2. This will introduce an error area alongside the running charge and the follow through with the manufacturer/designer support programs.

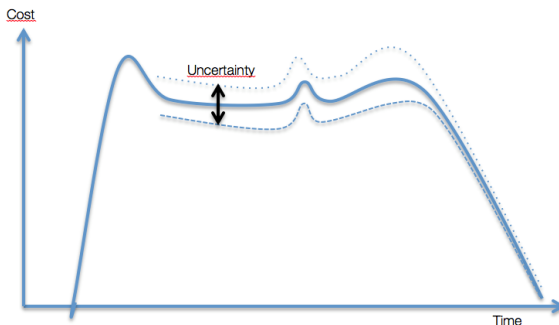


Fig. 4 Uncertainty in Life Cycle Cost with mid-life overhaul

Using all this data, it becomes possible to have a projection of the life cycle cost for each component and extrapolate it to the whole system. This is a crucial data at early design stages. It allows to focus on strategic sub-assemblies and/or design to a specific factor (maintenance, energy, ...) using 80/20 rule and life cycle cost.

5 Conclusion

This work presents a novel approach using projected life cycle cost profiles estimation for complex products (products having a prominent support cost and requires high availability) design process. This method is based on the usage of reliability for estimating life cycle costs.

The current modeling using only reliability, charge and expected lifespan is lacking the obsolescence and technological maturity that will impact life cycle cost. Macroeconomics factors in the region of installation (and support) also need to be taken into account.

This approach is expected to reduce operation costs by allowing better prediction of the operation costs, as well as manufacturing and disposal.

This work is currently being tested and should be finished mid-2013.

References

- [1] Clark, K.B., Fujimoto, T.: *Product development performance: strategy, organization, and management in the world auto industry*. Harvard Business School Press, Boston (1991)
- [2] Davies, A., Hodbay, M.: *Business of projects: managing innovation in complex products and systems*. Cambridge University Press, Cambridge (2011)
- [3] Keys, L.K.: System life cycle engineering and DF'X'. *IEEE Transactions on Components, Hybrids, and Manufacturing Technology* 13(1), 83–93 (1990)
- [4] Kriwet, A., Zussman, E., Seliger, G.: Systematic integration of design-for-recycling into product design. *International Journal of Production Economics* 38(1), 15–22 (1995)
- [5] Garin, T.A.: *A Preferred Spare Decision Support System Incorporating a Life Cycle Cost Model*. Air University (1991)
- [6] Blanchard, B.S., Fabrycky, W.J.: *Systems Engineering and Analysis*. International Series in Industrial & Systems Engineering (2010)
- [7] DoD 4100.35, *Development of integrated logistic support for systems and equipment*, U.S. Department of Defense (1964)
- [8] DoD 4245.3, *Design to Cost*, U.S. Department of Defense (1983)
- [9] Dangel, R.: *Integrated Logistic Support (ILS) Implementation in the Naval Ship System Command*. In: ASE 6th Annual Technical Symposium, pp. 1–25 (1969)
- [10] *Army Regulation 700–127, Integrated Logistics Support*, Washington DC, Headquarters Department of the Army (2007)
- [11] *Mil Std 785, Reliability Programs for Systems and Equipment, Development and Production*, National Technical Information Services, Springfield, Va
- [12] MIL-STD-1338-1A, *Logistics Support Analysis*
- [13] MIL-STD-1338-2B, *DoD Requirements for a Logistic Support Analysis Record*
- [14] Carrubba, E.R., et al.: *Intergrating Life-Cycle Cost and Cost-of-Ownership in the Commercial Sector*. In: *Annual Reliability and Maintainability Symposium*. IEEE (1992)
- [15] Dougan, K.W., Reilly, M.C.: *Quantitative reliability methods improve plant uptime*. *Hydrocarbon Processing*, 131–141 (1993)

- [16] Hokstad, P., et al.: Life Cycle Cost Analysis in Railway Systems. In: SINTEF Safety and Reliability (1998)
- [17] Rose, J., Phelps, E.: Cost of Ownership Application to Airplane Design. In: IEEE Annual Reliability and Maintainability Symposium, pp. 47–50. IEEE (1979)
- [18] de la Vega, F.F., et al.: Plant Reliability Analysis in LNG Plants. In: Proceedings of 11th International Conference on LNG, paper#2.11 (1995)
- [19] Leitz, C.W.: Life cycle cost modeling of automotive paint systems, Massachusetts Institute of Technology (2007)
- [20] Grabowski, H., Geiger, K.: Neue Wege zur Produktentwicklung, Karlsruhe, Karlsruhe Univ. (T.H.) (Germany). Inst. fuer Rechneranwendung in Planung und Konstruktion (1991)
- [21] Niazi, A., Dai, J.S., et al.: Product cost estimation: Technique classification and methodology review. *Journal of Manufacturing Science and Engineering* 128(2), 563–575 (2006)
- [22] Baguley, P., Maropoulos, P.G., Pease, R.: A roadmap for cost engineering in the automotive supplier sector. *International Journal of Manufacturing Technology and Management* 15(3), 265–283 (2008)
- [23] Newnes, L.B., Mileham, A.R., et al.: Predicting the whole-life cost of a product at the conceptual design stage. *Journal of Engineering Design* 19(2) (2006)

Verification of Infocommunication System Components for Modeling and Control of Saturated Traffic in Megalopolis^{*}

Marina V. Yashina and Andrew V. Provorov

Moscow Technical University of Communication and Informatics,
8-a Aviamotornaya str., Moscow, Russia
{yash-marina,ytrey}@yandex.ru

Abstract. The article contains the results of experimental research on testing and verification of the elements of infocommunication system, developed in the Scientific and Educational Center on Intelligent Monitoring and Transport Control Systems of Moscow Technical University of Communication and Informatics (MTUCI) and Mathematical Modeling Department of Moscow State Automobile & Road Technical University (MADI) for modeling and control of saturated traffic metropolis (The head is Professor A.P. Buslaev) [1]. As a result of the high rates of automobilization in the world are actual problems of modeling and forecasting of the traffic flow in complex networks of megalopolises. Actively develops the theory of traffic flows, as well as the means of observation and monitoring of the characteristics of the transport flows. But theoretical models of traffic significantly required of real information about the parameters and characteristics of such a complex and constantly changing socio-technical system, which is the transportation system of the megalopolis, including urban road network and traffic flows. Experiments carried out on the basis of the mobile laboratory of its own design, equipped with a special system of monitoring and processing on the PC and smartphones.

1 Introduction

Automobilization of megalopolises has reached a point, when the level of traffic congestion and increased gas pollution of the atmosphere is palpable to all residents, even those who are not a driver. Undoubtedly, the transport system of a large city is one of difficult socio-technical systems of the modern world. Owing to technological progress, complexity of the technical component of this system doesn't cause questions. Participation in it of a huge number of users, from pedestrians and drivers to officials and urban managers, makes a very high social significance of the system. It is possible to tell that all inhabitants of the megalopolis

^{*} This work has been supported by the Russian Foundation for Basic Research, grant No. 11-07-00622a.

are interested in overall performance of this system, even those who don't leave their apartments, breathe air, the main polluter of which the specialists unanimously recognize the automobile transport. Unstable movement on urban road network is characterized by sporadic resulting congestion, a total violation of the rules of the road, a large number of traffic accidents, environmental pollution and other negative indicators (factors).

The problem of modeling and control of traffic flows led to active development of the theory traffic flows, a significant contribution to the development of which was made by scientists all over the world. It is recognized, that the beginning of the development of the theory found in works B.D. Greenshields [2]. Among the classical approaches to the simulation of traffic, we note the classic car-following model, hydrodynamic approaches in works Lighthill-Whitham [3,4], research C.F. Daganzo [5,6], Nagel-Schreckenberg stochastic models [7,8].

Since 1990 of the 20th century Mathematical Modeling Department of MADI (under the supervision of Professor A.P. Buslaev) is engaged in complex research of simulation problems of traffic flows in megalopolises on the example of Moscow. Among the main achievements we would like to note the creation of hydrodynamic simulation models of Moscow traffic flows, creation and development of a deterministic-stochastic approach to the traffic modeling [9,10,11,12,13]. At present, in the Scientific and Educational Center under the leadership of professor A.P. Buslaev there is an association of young researchers of MTUCI and MADI.

Theoretical models are in need of determining the experimentally obtained data, parameters and constants of equations of dynamical systems. In developed countries, such as USA and Germany there is the intelligent transport system of monitoring traffic, and scientists have access to data, for example Kerner [14]. In our country such systems are absent or data are closed.

We've developed our own monitoring system, which can be extended on stationary and mobile points of traffic monitoring.

The most modern and perspective approach is based monitoring on the basis of mobile gadgets network. The results of approbation of this system will be shown in the following sections.

2 System Architecture

Results of verification of SSSR-traffic system [19,20] developed within the project [1] are presented in this article.

2.1 Hardware

Experiments carried out on the basis of the mobile laboratory¹, - Volkswagen Transporter (Fig. 1) with use of infocommunication devices - smartphones.

¹ The mobile laboratory is created within performance of the RFBR grant No. 08-01-01802-ex-b.



Fig. 1 Mobile laboratory on the basis of the Volkswagen Transporter (Property of MADI)

Technical characteristics of smartphone:

- Standard: GSM/GPRS/EDGE/UMTS/HSDPA 850/900/1800/1900/2100
- Screen: display 3.0"-4.0", the display resolution: 800 x 480
- Clock rate: over 1000 MHz, multi-core
- RAM: 512-1024 MB
- ROM: 512-1024 MB
- Data transfer: Bluetooth, USB, Wi-Fi (IEEE 802.11 b/g/n)
- Camera: 3.2-8 Mpx

To connect the smartphone to the network used a personal computer HP Proliant jh453.

2.2 *Software*

The server under control of Windows Server 2008 operational system. On it is installed the Internet Information Service (IIS). The server is responsible for communication with clients of a distributed system and the exchange of information with them.

- Smartphones under control of operating systems: Windows Mobile, Google Android.
- Development environments: Microsoft Visual Studio, Eclipse.
- Programming languages: C#, Java
- Web application framework: ASP.NET MVC (+AJAX)
- Communications protocols: TCP/IP, UDP, FTP.
- Data exchange formats: SOAP, JSON
- Database management system (DBMS): MySQL

The system is based on three-level (three-tier) architecture [17,18]. It presupposes the existence of the following application components: the client application connected to the application server which in turn is connected to the database server. Place of any of these components in the system can take the software to any vendor (Google, Apple, Windows, etc.).

3 Problem of Road Capacity Monitoring by Means of Vehicle GPS Tracks

Urban road networks of the megalopolis being represented by complex graph with a large number of nodes and edges. For models of flow on the traffic network it is important to know the real capacity of sites of the network.

Because of the randomly parked cars especially in the central areas of the city the bottlenecks are formed.

Monitoring of dynamically changing of road capacity is a difficult but important element of modeling of flows in the city. For the solution of this problem the data collection system in real time on the basis of a network of smartphones is created.

The driver of the control vehicle moves on a street road network. The car is equipped with the smartphone (Fig. 2) with the developed application (Fig. 3). By means of the application information is collected:

- latitude, longitude, time, instant speed (automatically);
- number of lanes in current section of road (manually);
- lane number on which the vehicle is at the moment (manually);
- numbers of lanes which are blocked for movement at the current time and place (parking, road accident etc.) (manually).

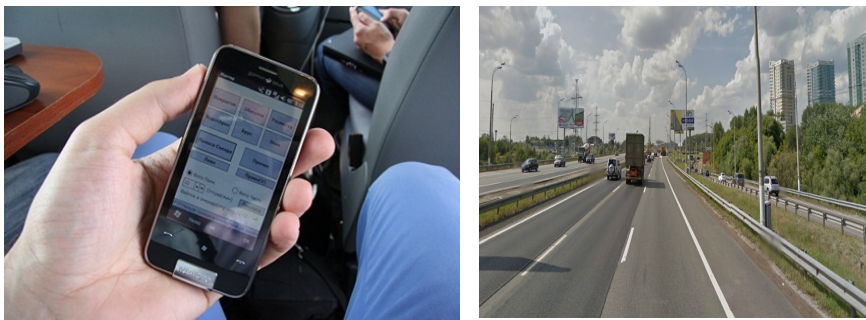


Fig. 2 Smartphone application at the experiment on the urban network

It is supposed that the control car moves according to traffic regulations and with the speed which the condition of traffic flow allows.

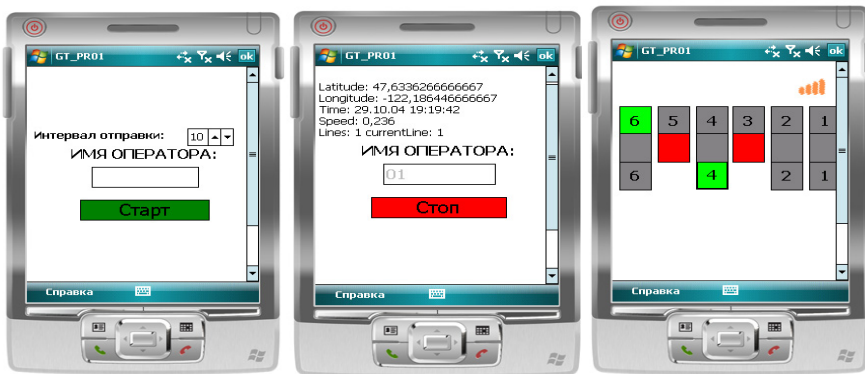


Fig. 3 Program interface

On the Settings tab, we can:

1. Enter the name of the operator and collect statistics;
2. Choose a sending interval (frequency with which the program sends data to the server);
3. Graphically see the level of GPS signal;
4. Run the program, click Start.

On the Lanes tab, we can:

1. In the top row - select the number of lanes on the road;
2. In the middle row - select the lane, the movement on which isn't carried out or difficult;
3. In the bottom row - select the current lane.

The program is multi-threading:

1. The main thread - graphical interface;
2. The thread of sending data to the server.

3.1 Experimental Results

Statistical data are stored on the smartphone in a format:

1. Operator;
2. Latitude;
3. Longitude;
4. Date/time;
5. Instant velocity;
6. Altitude
7. Number of lanes in current section of road;
8. Lane number on which the car is at the moment;

- 9. Numbers of lanes which are blocked for movement at the current time and place (parking, road accident etc.);
- 10. Number of satellites.

Then the data files are automatically sent to the server via the FTP protocol.

(1)	(2)	(3)	(4)	(5)	(6)	(7)(8)	(9)	(10)
01	55,79279415	37,54827851	08.11.11 6:33:29	143,3	1154,8	3 3	0	6(6)
01	55,79273087	37,54846612	08.11.11 6:33:30	51,8	163,4	3 3	0	6(6)
01	55,79264322	37,54863624	08.11.11 6:33:31	55,5	160,9	3 3	0	6(6)
01	55,79256128	37,5488153	08.11.11 6:33:32	55,5	160,7	3 3	0	6(6)
01	55,79247822	37,54899828	08.11.11 6:33:33	54,8	169,7	3 3	1	6(6)
01	55,7924526	37,54915225	08.11.11 6:33:34	52,4	162,1	3 3	1	6(6)
01	55,79237077	37,54933995	08.11.11 6:33:35	49,4	161,9	3 3	1	6(6)
01	55,79229719	37,54950372	08.11.11 6:33:36	47,5	163,8	3 3	1	5(6)
01	55,79221506	37,54966866	08.11.11 6:33:37	45,3	165,1	3 3	1	5(6)

Fig. 4 An example of data obtained on the smartphone and transferred to the server

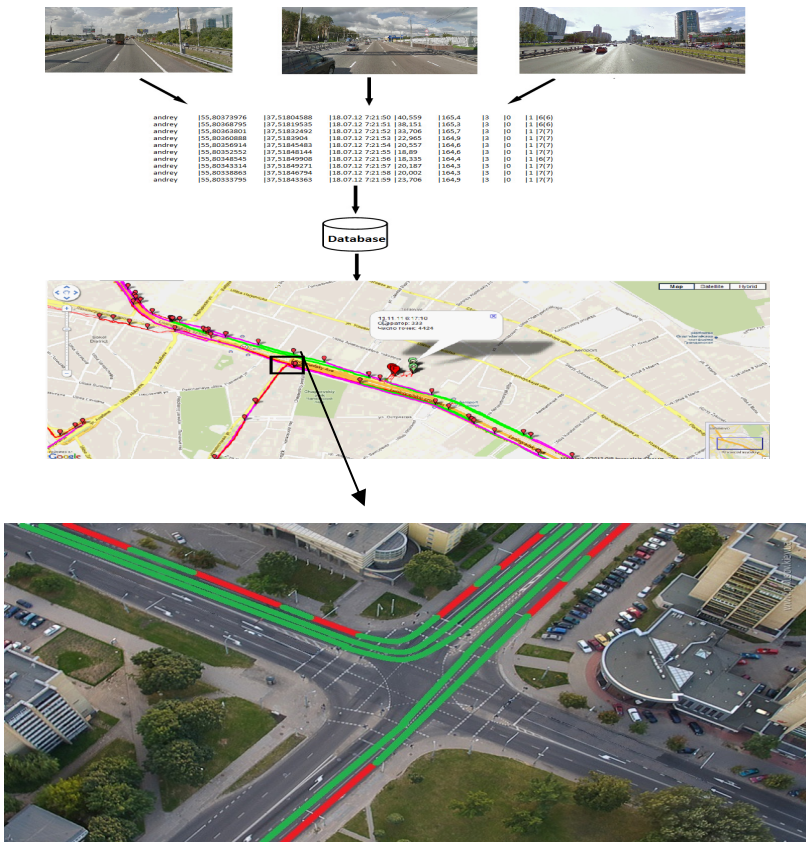


Fig. 5 Road capacity monitoring

4 The Registration System of Violations of the Dynamic Dimension

In the model of the leader-following the main measured parameters are the moving velocity and the distance between moving objects. At the same time, in traffic the speed limits are determined as driven "by eye", from the experience and understanding of a safe distance, which is often the cause of accidents. Creation of smart devices for the automobiles for safety, effectiveness and comfort during the trip is the most actual problem today.

We analyze the sequence of frames which are received from a position of driven during the leader following. To estimate the dynamics of the distance to the leading one, it is generated:

1. two virtual mutually perpendicular rectangular detectors which allow to analyze of image intensity signals and define the image boundaries of the leading one. Image size of leader, in its turn, allows to estimate the distance.
2. three detectors in the form of a trapeze. The central trapeze allows to estimate distance on all lanes, instead of on the center, unlike point 1). Two lateral trapezes allow to register vehicles on the adjacent lanes.

Created two desktop-applications with which help, algorithms for mobile OS are tested.

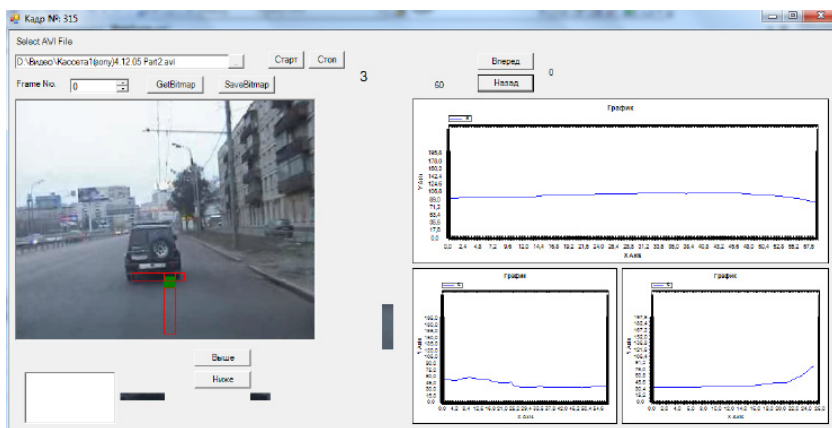


Fig. 6 The interface of desktop-application

Fig. 6 shows the interface of the program. Three scanners correspond to the three graph of the intensity distribution of the gray in these areas. The central scanner allows to estimate the distance to the vehicle ahead. Two side - horizontal boundaries of the car.

4.1 Calibration Applications

The parameters n_1 , n_2 , Δ_1 , Δ_2 , k_1 , k_2 , g are defined in the program settings, need its launch (Fig. 7). By default, $n_1=10$, $n_2=10$, $\Delta_1=18$, $\Delta_2=18$, $k_1=6$, $k_2=6$, $l=30$, $g=25$ for applications with rectangular detectors and $n_1=300$, $\Delta_1=5$, $\Delta_2=10$, $k_1=6$, $k_2=6$, n_2 (set with the mouse) for an application with the detectors in the form of trapezoids.

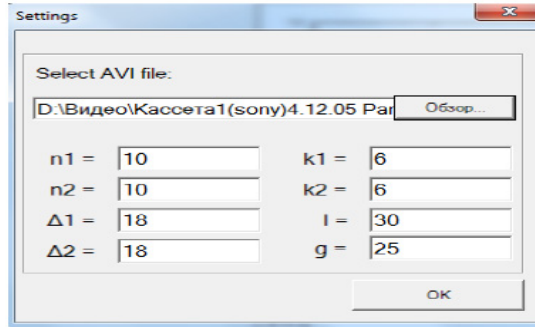


Fig. 7 Program settings

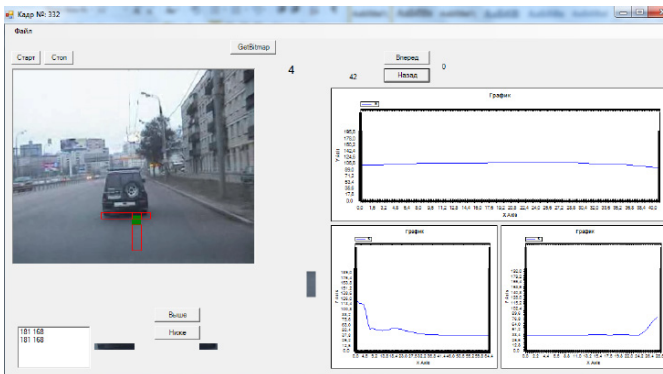


Fig. 8 The interface of desktop-application

In desktop-application video stream is loaded from a file and it is a storyboard. In application for a smartphone, the video stream received from the camera. Shows a dynamic dimension and is calculated by the formula:

$$d(v) = a + bV + cV^2,$$

where d - distance (m), V - velocity (m / s), $a = 5.7$, $b = 0.504$, $c = 0.0285$;

Thus, on a smartphone, you can compare the dynamic dimension received on a formula with the real dynamic dimension. In case of danger there will be a sound notification.

4.2 Multilane Estimation of the Dynamic Dimension and Multilane Traffic Safety

At unnormalized movement on multilane road for a safety estimation it is necessary to consider behavior of cars on the next lanes. In this case software generalization is developed. It is used three detectors in the form of a trapeze. The central trapeze allows to estimate distance on all lanes, instead of on the center, unlike rectangular detectors. Two lateral trapezes allow to register vehicles on the adjacent lanes.

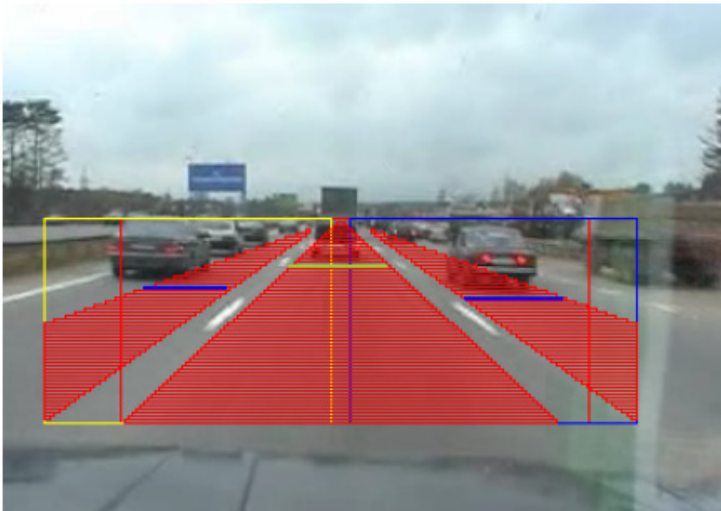


Fig. 9 The scope of analysis and detectors in the form of a trapeze

Fig. 9 shows a screenshot of the program. The user clicks indicates the line of the horizon and the focus. Each of the three detectors (trapeze) are inscribed in the appropriate boxes. As the movement of the vehicle, the width of the rectangles is automatically adjusted for the layout of the road and selected so as to best reach the traffic lanes. The numbers are displayed distance to the vehicle ahead respectively for the three lanes.

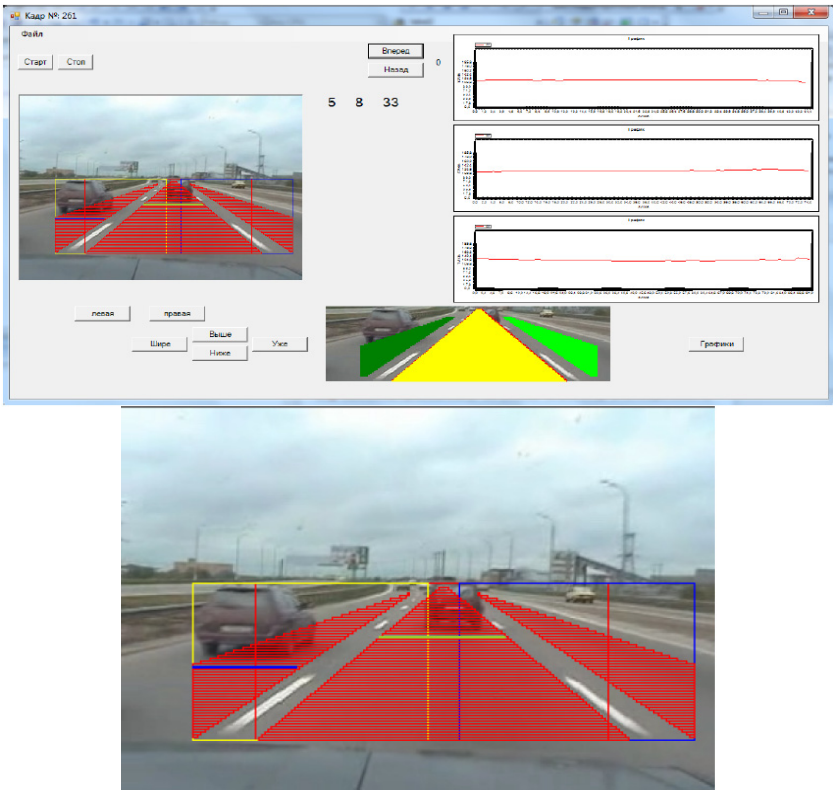


Fig. 10 Program interface

Fig. 10 shows the calculation of the dynamic dimension to the vehicle flow.

4.3 Testing and Calibration Applications

Thus, installed on a particle (flow component), a smartphone with application (Fig. 11), can control the area of the dynamic dimension and in case of danger situations record and send a signal to the server and other parts of infocommunication systems.



Fig. 11 Smartphone with the application

5 Conclusions and Future Work

The paper presents the results of experiments in real conditions on saturated flow on Moscow traffic network. We tested several systems related to micro and macro levels.

Scenario of the «road capacity monitoring» problem, presented in section 3 and implemented by SSSR-traffic system, provides a unique opportunity to receive information the real network usage over time and space. Data is collected by labeled cars in saturated flows. The server processes the accumulated statistics and creates mean prediction.

The scenario of 4-th section problem, measured of dynamic dimension, allows to analyze the traffic flows characteristics and use the adequate flow models of theory of dynamical systems.

Further development of the system is seen to build relationships between the individual components. Planned to form an integrated project to model and predict flows metropolis based on the system with the capture of information from mobile devices.

Acknowledgments. We are grateful to Professor A.P. Buslaev for the problem statement and discuss the results. The research presented in this work has been supported by the Russian Foundation for Basic Research, grant No. 11-07-00622a.

References

- [1] Report of Project GK № 14.740.11.0397, Ministry of Education of RF. Supervisor prof. A.P. Buslaev. Theoretical and applied problems of development of intelligent system for monitoring and control of distributed processes. - M. MTUCI (2012)
- [2] Greenshields, B.D.: A study of traffic capacity. Proceed. US Highway Research Board 14, 448–494 (1934)
- [3] Whitham, G.B.: Linear and Nonlinear Waves. John Wiley & Sons (1974)
- [4] Lighthill, M.J., Whitam, G.B.: On kinematic waves II. Theory of Traffic Flow on long Growned Roads. Proc. Roy. Soc. A 229(1178), 317–345
- [5] Dangazo, C.F.: The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. Transp. Res. B 28(4), 269–287 (1994)
- [6] Dangazo, C.F.: The cell transmission model, Part II: Network traffic. Transp. Res. B 29(2), 79–93 (1995)
- [7] Nagel, K., Schreckenberg, M.: A cellular automation model for freeway traffic. Phys. I France 2, 2221–2229 (1992)
- [8] Nagel, K.: Particle hopping models and traffic flow theory. Physical review E 51(5), 4655 (1996)
- [9] Lukanin, V.N., Buslaev, A.P., Trofimenko, Y.V., Yashina, M.V.: Traffic Flows and Environment, Moscow, INFRA-M, 408 p. (1998); Lukanin, V.N., Buslaev, A.P., Yashina, M.V.: Traffic Flows and Environment – 2, Moscow, INFRA-M, 644 p. (2001) (in Russian)

- [10] Buslaev A.P., Novikov, Prikhodko, V.M., Tatashev, A.G., Yashina M.V.: Stochastic and simulation approach to traffic, p. 286. Mir, Moscow (2003) (in Russian)
- [11] Kozlov, V.V., Buslaev, A.P.: Metropolis Traffic Modeling: from Intelligent Monitoring through Physical Representation to Mathematical Problems. In: Proc. of Int. Conf. CMMSE 2012, vol. 1, pp. 750–756 (2012)
- [12] Kozlov, V.V., Buslaev, A.P., Tatashev, A.G.: Monotonic random walks and cluster flows on networks. Models and traffic applications. LAP LAMBERT, 300 p. Academic Publ. (2013)
- [13] Buslaev, A.P., Dorgan, V.V., Kuzmin, D.M., Prikhodko, V.M., Travkin, V.U., Yashina, M.V.: Recognition of Images and Monitoring of Road Conditions, Traffic Flows and Safety. J. Vestnik MADI (4), 102–109 (2005) (in Russian)
- [14] Kerner, B.S.: Introduction to modern traffic flow theory and control. Springer, Berlin (2009)
- [15] Buslaev, A.P., Provorov, A.V., Yashina, M.V.: Traffic and Distributed Information-Calculated Networks. Problems and Solutions. Part 1. Traffic and Positioning. - M., Tech-PoligrafCenter, pp. 1–263 (2011) (in Russian)
- [16] Buslaev, A.P., Provorov, A.V., Yashina, M.V.: Traffic and Distributed Information-Calculated Networks. Problems and Solutions. Part 2. Pattern Recognition. - M., TechPoligrafCenter, pp. 1–108 (2011) (in Russian)
- [17] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services - Concepts, Architectures and Applications. Springer (2004)
- [18] Ramirez, A.O.: Three-Tier Architecture. Linux Journal (July 1, 2000)
- [19] Provorov, A.V.: On algorithms and software for traffic intelligent systems using SSSR mobile devices system. In: Proc. of Int. Conf. CMMSE 2012, vol. 2, pp. 976–981 (2012)
- [20] Provorov, A.V.: Intellectual problems of traffic and SSSR-system. Journal Vestnik MADI (1), 79–85 (2013) (in Russian)

Shuffle-Based Verification of Component Compatibility

W.M. Zuberek

Department of Computer Science, Memorial University,
St.John's, NL, Canada A1B 3X5

Abstract. An extension of earlier work on component compatibility is described in this paper. Similarly as before, the behavior of components is specified by component interface languages, and the shuffle operation is introduced to represent possible interleavings of service requests that originate at several concurrent components. The paper shows that the verification of component compatibility is possible without the exhaustive analysis of the state space of interacting components. Exhaustive analysis of state spaces was the basis of earlier approaches to compatibility verification.

Keywords: software components, component-based systems, component composition, component compatibility, compatibility verification, shuffle operation, labelled Petri nets.

1 Introduction

In component-based systems [8], two interacting components, one requesting services and the other providing them, are considered compatible if all possible sequences of services requested by one component can be provided by the other one. This concept of component compatibility can be extended to sets of interacting components, however, in the case of several requester components, as is typically the case of client-server applications, the requests from different components can be interleaved and then verifying component compatibility must take into account all possible interleavings of requests. Such interleaving of requests can lead to unexpected behavior of the composed system, e.g. a deadlock can occur [17], [18].

The behavior of components is usually described at component interfaces [14] and the components are characterized as requester (active) and provider (reactive) components. Although several approaches to checking component composability have been proposed [1], [2], [3], [9], [11], [15], further research is needed to make these ideas practical [7].

The paper is an extension of previous work on component compatibility and substitutability [6], [16], [17], [18]. Using the same formal specification of component behavior in the form of interface languages defined by labeled Petri nets, the paper

extends the linguistic approach to the verification of component compatibility. A shuffle operation is proposed to represent the interleavings of requests originating at concurrent components. This shuffle of requests is matched with the (interface) language of the service provider. If the provider languages matches the interleavings of the requester components, the components are considered compatible, otherwise some correcting procedure is required (in the form of redesign of some components or additional constraints which preventing some interleavings to happen).

Since interface languages are usually infinite, their compact finite specification is needed for effective verification, comparisons and other operations. Labeled Petri nets [16], [17] are used as such specification.

Petri nets [12], [13] are formal models of systems which exhibit concurrent activities with constraints on frequency or orderings of these activities. In labeled Petri nets, labels, which represent services, are associated with elements of nets in order to identify interacting components. Well-developed mathematical theory of Petri nets provides a convenient formal foundation for analysis of systems modeled by Petri nets.

Section 2 introduces the shuffle operation applied to sequences of requests as well as collections of such sequences. Interface languages as the description of component's behavior are recalled in Section 3, while Section 4 provides the linguistic version of component compatibility. Section 5 illustrates the shuffle-based verification of components compatibility and Section 6 concludes the paper.

2 Shuffle and Swap

The shuffle operation, used in mathematical linguistics [10] to merge strings, is used here to represent the interleaving of requests from several components. So, if x_i and x_j are sequences of requests from components “ i ” and “ j ”, then $\text{shuffle}(x_i, x_j)$ denotes the set of sequences of merged requests in which all elements of x_i and x_j occur in their original order, but the elements of x_i can be arbitrarily interleaved with the elements of x_j . For example:

$$\text{shuffle}(ab, cd) = \{abcd, acbd, acdb, cabd, cadb, cadb, cdab\}.$$

The set of “shuffled” strings can also be created by successive applications of a swap operation to the concatenated string xy , where each swap operation “swaps” (changes the positions of) two adjacent symbols provided one of these two symbols is an element of x and the other is an element of y , so a simple swap (sswap):

$$\text{sswap}(abcd) = \{acbd\}$$

while the consecutive simple swap operation can be applied in three different ways, so:

$$\text{sswap}(acbd) = \{cabd, abcd, acdb\}$$

Let a (general) swap operation be the reflexive, transitive closure of the simple swap operation. Then:

$$\text{shuffle}(x,y) = \text{swap}(xy).$$

The shuffle operation can be naturally extended to sets of sequences:

$$\text{shuffle}(A,B) = \{\text{shuffle}(x,y) \mid x \in A \wedge y \in B\}.$$

Moreover, it can be observed that:

$$\text{shuffle}(x, \text{shuffle}(y,z)) = \text{shuffle}(\text{shuffle}(x,y), z)$$

so the operation can be generalized as:

$$\text{shuffle}(x,y,z,\dots)$$

as well as:

$$\text{shuffle}(A,B,C,\dots).$$

3 Component Behavior

The behavior of a component, at its interface, can be represented by a cyclic labeled Petri net [5], [6], [17]:

$$\mathcal{M}_i = (P_i, T_i, A_i, S_i, m_i, \ell_i, F_i),$$

where P_i and T_i are disjoint sets of places and transitions, respectively, A_i is the set of directed arcs, $A_i \subseteq P_i \times T_i \cup T_i \times P_i$, S_i is an alphabet representing the set of services that are associated with transitions by the labeling function $\ell_i : T_i \rightarrow S_i \cup \{\varepsilon\}$ (ε is the “empty” service; it labels transitions which do not represent services), m_i is the initial marking function $m_i : P_i \rightarrow \{0, 1, \dots\}$, and F_i is the set of final markings (which are used to capture the cyclic nature of sequences of firings).

Sometimes it is convenient to separate net structure $\mathcal{N} = (P, T, A)$ from the initial marking function m .

In order to represent component interactions, the interfaces are divided into *provider* interfaces (or p-interfaces) and *requester* interfaces (or r-interfaces). In the context of a provider interface, a labeled transition can be thought of as a service provided by that component; in the context of a requester interface, a labeled transition is a request for a corresponding service. For example, the label can represent a conventional procedure or method invocation. It is assumed that if the p-interface requires parameters from the r-interface, then the appropriate number and types of parameters are delivered by the r-interface. Similarly, it is assumed that the p-interface

provides an appropriate return value, if such a value is required. The equality of symbols representing component services (provided and requested) implies that all such requirements are satisfied.

For unambiguous interactions of requester and provider interfaces, it is required that in each p-interface there is exactly one labeled transition for each provided service:

$$\forall t_i, t_j \in T : \ell(t_i) = \ell(t_j) \neq \varepsilon \Rightarrow t_i = t_j.$$

Moreover, to express the reactive nature of provider components, all provider models are required to be ε -conflict-free, *i.e.*:

$$\forall t \in T \forall p \in \text{Inp}(t) : \text{Out}(p) \neq \{t\} \Rightarrow \ell(t) \neq \varepsilon$$

where $\text{Out}(p) = \{t \in T \mid (p, t) \in A\}$; the condition for ε -conflict-freeness could be used in a more relaxed form but this is not discussed here for simplicity of presentation.

Component behavior is determined by the set of all possible sequences of services (required or provided by a component) at a particular interface. Such a set of sequences is called the *interface language*.

Let $\mathcal{F}(\mathcal{M})$ denote the set of firing sequences in \mathcal{M} such that the marking created by each firing sequence belongs to the set of final markings F of \mathcal{M} . The interface language $\mathcal{L}(\mathcal{M})$, of a component represented by a labeled Petri net \mathcal{M} , is the set of all labeled firing sequences of \mathcal{M} :

$$\mathcal{L}(\mathcal{M}) = \{\ell(\sigma) \mid \sigma \in \mathcal{F}(\mathcal{M})\},$$

where $\ell(t_{i_1} t_{i_2} \dots t_{i_k}) = \ell(t_{i_1}) \ell(t_{i_2}) \dots \ell(t_{i_k})$.

By using the concept of final markings, interface languages can easily capture the cyclic behavior of (requester as well as provider) components.

Interface languages defined by Petri nets include regular languages, some context-free and even context-sensitive languages [10]. Therefore, they are significantly more general than languages defined by finite automata [4], but their compatibility verification is also more difficult than in the case of regular languages.

4 Component Compatibility

Interface languages of interacting components are used to define the compatibility of components. For a pair of interacting components, a requester component “ r ” and a provider component “ p ” are compatible if and only if all sequences of services requested by “ r ” can be provided by “ p ”, *i.e.*, if and only if:

$$\mathcal{L}_r \subseteq \mathcal{L}_p.$$

In the case of several requester components, indicated by subscripts “ $i \in I$ ” where I is an index set, interacting with a single provider component “ p ”, the component compatibility requires that all sequences of (interleaved) requests be satisfied by the provider, so in a straightforward case:

$$\text{shuffle}(\mathcal{L}_i | i \in I) \subseteq \mathcal{L}_p.$$

Often however, some requests cannot be satisfied when they are requests and are delayed because some other operations performed by the provider component. In such cases the services can be provided in a sequence which is different from the sequence of requests. Therefore, it is convenient to decompose the sequence of requests x in two parts y and z , $x = yz$, the initial part y which is served in the order of requests, and the remaining part z where the services are provided in an order different than requested. And then the component compatibility condition is:

$$\forall x \in \text{shuffle}(\mathcal{L}_i | i \in I) : x \in \mathcal{L}_p \vee x = yz \wedge \{y\} \circ \text{swap}(z) \cap \mathcal{L}_p \neq \emptyset$$

where \circ denotes set concatenation, and \emptyset is the empty set.

5 Example

A simple system of two requesters and a single provider is shown in Fig.1 [18]. The interface language of the provider is described by a regular expression:

$$\mathcal{L}_p = ((ab + ba)c)^*$$

and the language of the (interleaved) requests from two requesters is:

$$\mathcal{L}_r = (\text{shuffle}(abc, bac))^*.$$

Because of cyclicity of interface languages, the length of analyzed sequences can be restricted to 6, and than:

$$\mathcal{L}_p^{(6)} = \{abcabc, abcbac, bacabc, bacbac\}$$

and there are 10 different strings in $\mathcal{L}_r^{(6)}$:

$$\mathcal{L}_r^{(6)} = \{ababcc, abacbc, abbacc, abbcac, abcbac, \\ babacc, babcac, baabcc, baacbc, bacabc\}.$$

Verification of the component consistency checks the sequences of requests in $\mathcal{L}_r^{(6)}$ (in the following table, symbol subscripts indicate the original component requesting the service) looking for a matching sequence in $\mathcal{L}_p^{(6)}$:

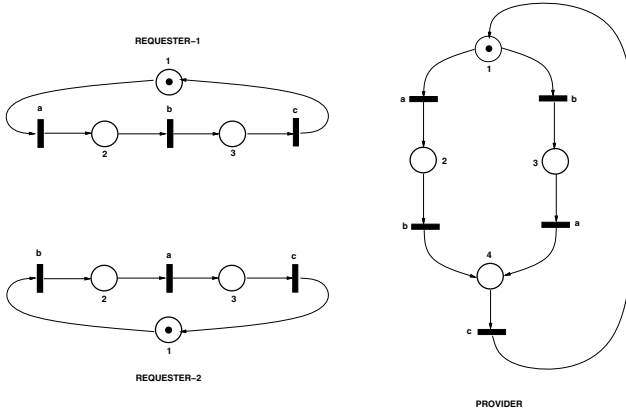


Fig. 1 Two requesters and a single provider

$x \in \mathcal{L}_r^6$	matching in $\mathcal{L}_p^{(6)}$
$a_1b_2a_2b_1c_1c_2$	$\{ab\} \circ \text{swap}(a_2b_1c_1c_2) \cap \mathcal{L}_p^{(6)} = \emptyset$
$a_1b_2a_2b_1c_2c_1$	$\{ab\} \circ \text{swap}(a_2b_1c_2c_1) \cap \mathcal{L}_p^{(6)} = \emptyset$
$a_1b_1b_2a_2c_1c_2$	$cbac \in \text{swap}(b_2a_2c_1c_2)$ and $ab \circ cbac \in \mathcal{L}_p^{(6)}$
$a_1b_1b_2a_2c_2c_1$	$cbac \in \text{swap}(b_2a_2c_2c_1)$ and $ab \circ cbac \in \mathcal{L}_p^{(6)}$
$a_1b_2b_1a_2c_1c_2$	$\{ab\} \circ \text{swap}(a_2b_1c_1c_2) \cap \mathcal{L}_p^{(6)} = \emptyset$
$a_1b_2b_1a_2c_2c_1$	$\{ab\} \circ \text{swap}(a_2b_1c_2c_1) \cap \mathcal{L}_p^{(6)} = \emptyset$
$a_1b_1b_2c_1a_2c_2$	$cbac \in \text{swap}(b_2c_1a_2c_2)$ and $ab \circ cbac \in \mathcal{L}_p^{(6)}$
$a_1b_2b_1c_1a_2c_2$	$\{ab\} \circ \text{swap}(a_2b_1c_1c_2) \cap \mathcal{L}_p^{(6)} = \emptyset$
$a_1b_1c_1b_2a_2c_2$	$abcbac \in \mathcal{L}_p^{(6)}$
$b_2a_1b_1a_2c_1c_2$	$\{ba\} \circ \text{swap}(b_1a_2c_1c_2) \cap \mathcal{L}_p^{(6)} = \emptyset$
$b_2a_1b_1a_2c_2c_1$	$\{ba\} \circ \text{swap}(b_1a_2c_2c_1) \cap \mathcal{L}_p^{(6)} = \emptyset$
$b_2a_2a_1b_1c_1c_2$	$cabc \in \text{swap}(a_1b_1c_1c_2)$ and $ba \circ cabc \in \mathcal{L}_p^{(6)}$
$b_2a_2a_1b_1c_2c_1$	$cabc \in \text{swap}(a_1b_1c_2c_1)$ and $ba \circ cabc \in \mathcal{L}_p^{(6)}$
$b_2a_1b_1c_1a_1c_2$	$\{ba\} \circ \text{swap}(b_1c_1a_1c_2) \cap \mathcal{L}_p^{(6)} = \emptyset$
$b_2a_1b_1c_2a_1c_1$	$\{ba\} \circ \text{swap}(b_1c_2a_1c_1) \cap \mathcal{L}_p^{(6)} = \emptyset$
$b_2a_2a_1c_2b_1c_1$	$cabc \in \text{swap}(a_1c_2b_1c_1)$ and $ba \circ cabc \in \mathcal{L}_p^{(6)}$
$b_2a_1a_2c_2b_1c_1$	$\{ba\} \circ \text{swap}(a_2c_2b_1c_1) \cap \mathcal{L}_p^{(6)} = \emptyset$
$b_2a_2c_2a_1b_1c_1$	$bacabc \in \mathcal{L}_p^{(6)}$

All rows which do not have a matching sequence indicate component incompatibilities.

For example, for the first sequence of requests, $x = a_1b_2a_2b_1c_1c_2$, x is decomposed into ab (which can be matched by the provider) and the remaining sequence $a_2b_1c_1c_2$, for which the swap operation creates the set:

$$\{a_2b_1c_1c_2, a_2b_1c_2c_1, a_2c_2b_1c_1, b_1a_2c_1c_2, b_1a_2c_2c_1, b_1c_1a_2c_2\}$$

or, removing the subscripts:

$$\{abcc, acbc, bacc, bcac\}$$

and then the intersection:

$$\{ab\} \circ \{abcc, acba, baccbcac\} \cap \mathcal{L}_p^{(6)}$$

is empty indicating the incompatibility.

The second sequence is basically identical, and so on.

To eliminate incompatibilities existing in this example, service renaming was proposed in [18], for example, the provider language can be (formally) changed from:

$$((ab + ba)c)^*$$

to:

$$((aB + bA)c)^*$$

by introducing services A and B as renamed services a and b, respectively, and changing the languages of the requesters in a similar way, to $(aBc)^*$ and $(bAc)^*$, respectively. After such renaming the interleavings of the two requesters are:

$$\text{shuffle}(aBc, bAc) = \{abABcc, abAcBc, abBAcc, abBcAn, aBbAcc, aBbcAc, aBcbAc, baBacc, baBcAc, baABcc, baAcBc, bAaBcc, bAacBc, bAcaBc\}$$

and the verification of the component compatibility follows the same steps as in the previous case. For example, the first sequence $abABcc$ can be decomposed into the leading a and the remaining $bABcc$ with the following set of swapped sequences:

$$\text{swap}(b_2A_2B_1c_1c_2) = \{b_2B_1A_2c_1c_2, b_2B_1c_1A_2c_2, B_1b_2A_2c_1c_2, B_1b_2c_1A_2c_2, B_1c_1b_2A_2c_2\}$$

Since, in this case::

$$\mathcal{L}_p^{(6)} = \{aBcaBc, aBcbAc, bAcaBc, bAcbAc\}$$

the compatibility is verified as:

$$a \circ \text{swap}(b_2A_2B_1c_1c_2) \cap \mathcal{L}_p^{(6)} = \{a_1B_1c_1b_2A_2c_2\} \neq \emptyset,$$

so the incompatibility has been removed.

It should be observed that the initial decomposition of the analyzed sequence is not necessary; its purpose is to simplify the verification process by reducing the length of the analysed sequence.

The remaining sequences are verified similarly, as shown in the following table.

$x \in \mathcal{L}_r^6$	matching in $\mathcal{L}_p^{(6)}$
$a_1b_2A_2B_1c_1c_2$	$BcbAc \in \text{swap}(b_2A_2B_1c_1c_2)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1b_2A_2B_1c_2c_1$	$BcbAc \in \text{swap}(b_2A_2B_1c_2c_1)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1b_2A_2c_2B_1c_1$	$BcbAc \in \text{swap}(b_2A_2c_2B_1c_1)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1b_2B_1A_2c_1c_2$	$BcbAc \in \text{swap}(b_2B_1A_2c_1c_2)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1b_2B_1A_2c_2c_1$	$BcbAc \in \text{swap}(b_2B_1A_2c_2c_1)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1b_2B_1c_1A_2c_2$	$BcbAc \in \text{swap}(b_2B_1c_1A_2c_2)$ and $a \circ BcbAc \in \mathcal{L}_p^{(6)}$
$a_1B_1b_2A_2c_1c_2$	$cbAc \in \text{swap}(b_2A_2c_1c_2)$ and $aB \circ cbAc \in \mathcal{L}_p^{(6)}$
$a_1B_1b_2A_2c_2c_1$	$cbAc \in \text{swap}(b_2A_2c_2c_1)$ and $aB \circ cbAc \in \mathcal{L}_p^{(6)}$
$a_1B_1b_2c_1A_2c_2$	$cbAc \in \text{swap}(b_2c_1A_2c_2)$ and $aB \circ cbAc \in \mathcal{L}_p^{(6)}$
$a_1B_1c_1b_2A_2c_2$	$aBcbAc \in \mathcal{L}_p^{(6)}$
$b_2a_1A_2B_1c_1c_2$	$AcaBc \in \text{swap}(a_1A_2B_1c_1c_2)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2a_1A_2B_1c_2c_1$	$AcaBc \in \text{swap}(a_1A_2B_1c_2c_1)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2a_1A_2c_2B_1c_1$	$AcaBc \in \text{swap}(a_1A_2c_2B_1c_1)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2a_1B_1A_2c_1c_2$	$AcaBc \in \text{swap}(a_1B_1A_2c_1c_2)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2a_1B_1A_2c_2c_1$	$AcaBc \in \text{swap}(a_1B_1A_2c_2c_1)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2a_1B_1c_1A_2c_2$	$AcaBc \in \text{swap}(a_1B_1c_1A_2c_2)$ and $b \circ AcaBc \in \mathcal{L}_p^{(6)}$
$b_2A_2a_1B_1c_1c_2$	$caBc \in \text{swap}(a_1B_1c_1c_2)$ and $bA \circ caBc \in \mathcal{L}_p^{(6)}$
$b_2A_2a_1B_1c_2c_1$	$caBc \in \text{swap}(a_1B_1c_2c_1)$ and $bA \circ caBc \in \mathcal{L}_p^{(6)}$
$b_2A_2a_1c_2B_1c_1$	$caBc \in \text{swap}(a_1c_2B_1c_1)$ and $bA \circ caBc \in \mathcal{L}_p^{(6)}$
$b_2A_2c_2a_1B_1c_1$	$bAcaBc \in \mathcal{L}_p^{(6)}$

In this case, all sequences of requests are matched by the provider component, so the components are compatible.

6 Concluding Remarks

The paper shows that the verification of component compatibility based on the exhaustive analysis of the “state space”, as discussed in [16] and [17], can be replaced by a simple analysis of languages that describe the sets of request sequences that can be generated by interacting components. In fact, once the interface languages are known, the behavioral models of components are not needed at all.

It is believed that the proposed verification of component compatibility can be quite efficient since many symmetries and partial orders can be taken into account. All these properties are not addressed in this paper.

It should be noticed that the discussion was restricted to a single provider component. In the case of several providers, each provider can be considered independently of other, so a single provider case is not really a restriction.

Also, an important aspect of component compatibility is its incremental verification. The approach described in this paper is not incremental but may provide a foundation for an incremental approach.

The paper did not address the question of deriving behavioral models of components (which is common to all component-based studies). Such models, at least theoretically, could be derived from formal component specifications, or perhaps could be obtained through analyzing component implementations. Since the component compatibility verification proposed in this paper does not require the use of the underlying component models (they are used only to define the interface languages), these interface languages could also be determined experimentally, by executing the components and collecting the information about the sequences of service requests.

Acknowledgement. The Natural Sciences and Engineering Research Council of Canada partially supported this research through grant RGPIN-8222.

References

1. Attiogbé, J.C., André, P., Ardourel, G.: Checking component composability. In: Löwe, W., Südholt, M. (eds.) SC 2006. LNCS, vol. 4089, pp. 18–33. Springer, Heidelberg (2006)
2. Baier, C., Klein, J., Klüppelholz, S.: Modeling and verification of components and connectors. In: Bernardo, M., Issarny, V. (eds.) SFM 2011. LNCS, vol. 6659, pp. 114–147. Springer, Heidelberg (2011)
3. Broy, M.: A theory of system interaction: components, interfaces, and services. In: *Interactive Computations: The New Paradigm*, pp. 41–96. Springer (2006)
4. Chaki, S., Clarke, S.M., Groce, A., Jha, S., Veith, H.: Modular verification of software components in C. *IEEE Trans. on Software Engineering* 30(6), 388–402 (2004)
5. Craig, D.C., Zuberek, W.M.: Compatibility of software components – modeling and verification. In: *Proc. Int. Conf. on Dependability of Computer Systems, Szklarska Poreba, Poland*, pp. 11–18 (2006)
6. Craig, D.C., Zuberek, W.M.: Petri nets in modeling component behavior and verifying component compatibility. In: *Proc. Int. Workshop on Petri Nets and Software Engineering, Siedlce, Poland*, pp. 160–174 (2007)
7. Crnkovic, I., Schmidt, H.W., Stafford, J., Wallnau, K.: Automated component-based software engineering. *The Journal of Systems and Software* 74(1), 1–3 (2005)
8. Garlan, D.: Formal modeling and analysis of software architecture: components, connectors, and events. In: Bernardo, M., Inverardi, P. (eds.) SFM 2003. LNCS, vol. 2804, pp. 1–24. Springer, Heidelberg (2003)
9. Henrio, L., Kammüller, F., Khan, M.U.: A framework for reasoning on component composition. In: de Boer, F.S., Bonsangue, M.M., Hallerstede, S., Leuschel, M. (eds.) FMCO 2009. LNCS, vol. 6286, pp. 1–20. Springer, Heidelberg (2010)
10. Hopcroft, J.E., Motwani, R., Ullman, J.D.: *Introduction to automata theory, languages, and computations*, 2nd edn. Addison-Wesley (2001)
11. Leicher, A., Busse, S., Süß, J.G.: Analysis of compositional conflicts in component-based systems. In: Gschwind, T., Aßmann, U., Wang, J. (eds.) SC 2005. LNCS, vol. 3628, pp. 67–82. Springer, Heidelberg (2005)

12. Murata, T.: Petri nets: properties, analysis, and applications. *Proceedings of the IEEE* 77(4), 541–580 (1989)
13. Reisig, W.: Petri nets – an introduction (EATCS Monographs on Theoretical Computer Science 4). Springer (1985)
14. Szyperski, C.: Component software: beyond object-oriented programming, 2nd edn. Addison-Wesley Professional (2002)
15. Zaremski, A.M., Wang, J.M.: Specification matching of software components. *ACM Trans. on Software Engineering and Methodology* 6(4), 333–369 (1997)
16. Zuberek, W.M.: Checking compatibility and substitutability of software components. In: *Models and Methodology of System Dependability*, Oficyna Wydawnicza Politechniki Wrocławskiej, ch. 14, pp. 175–186 (2010)
17. Zuberek, W.M.: Incremental composition of software components. In: Zamojski, W., Kacprzyk, J., Mazurkiewicz, J., Sugier, J., Walkowiak, T. (eds.) *Dependable Computer Systems*. AISC, vol. 97, pp. 301–311. Springer, Heidelberg (2011)
18. Zuberek, W.M.: Service renaming in component composition. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) *Complex Systems and Dependability*. AISC, vol. 170, pp. 319–330. Springer, Heidelberg (2012)

Author Index

- Aciu, Razvan 225
Aciu, Razvan-Mihai 1
Ahmed, Boubakeur 53
Améziane, Aoussat 523
- Bagiński, Jacek 11
Białas, Andrzej 25
Bluemke, Ilona 41
Boyarchuk, Artem 245
Brezhniev, Eugene 245
Buslaev, Alexander P. 63
Byłak, Michał 75
- Caban, Dariusz 83
Chrabski, Bartosz 93
Ciocarlie, Horia 1, 225
Czubak, Adam 109, 173
- Demkiewicz, Maciej 399
de Oliveira Rech, Luciana 365
Derezińska, Anna 119
Dettoni, Fernando 131
- Edifor, Ernest 141
- Falas, Lukasz 389
Frolov, Alexander 153
- Gola, Mariusz 173
Gorawski, Marcin 183
Gordon, Neil 141
Gorodnichev, M.G. 195
Grabski, Franciszek 205
- Hanandeh, Feras 215
- Istin, Codruta-Mihaela 225
- Józwiak, Ireneusz 237
Jureczek, Paweł 183
- Kamiński, Marek 501
Kędziora, Michał 237
Khaled, Benfriha 523
Kharchenko, Vyacheslav 245
Kijas, Szymon 255
Kokot, Adam 399
Kotas, Rafał 501
Krzykowska, Karolina 353
Kulesza, Karol 41
Kulesza, Zbigniew 501
- Larbi, Boukezzi 53
Lasek, Piotr 275
Laskowski, Dariusz 75, 295
Lubaś, Robert 285
Lubkowski, P. 295
Luiz, Aldelir Fernando 131
Lung, Lau Cheuk 131, 365
- Magnabosco, Leandro Quibem 365
Majzik, István 469
Marciniak, Paweł 501
Mazurkiewicz, Jacek 307
Melińska, Aleksandra 237
Miller, Janusz 285
Mycek, Marcin 285, 315
- Napieralski, Andrzej 501
Nigmatulin, A.N. 195
Nikodem, Maciej 439

- Orłowski, Cezary 93
 Pikulski, Wojciech 325
 Porzycki, Jakub 285
 Provorov, Andrew V. 531
 Quasmeh, Yaser 215
 Ratkowski, Andrzej 335
 Rosiński, Adam 353
 Sacha, Krzysztof 325
 Schauer, Patryk 399
 Shmakov, Anton 345
 Siergiejczyk, Mirosław 353
 Silva, Marcelo Ribeiro Xavier 365
 Sklyar, Vladimir 245
 Słabicki, Mariusz 439
 Stasiak, Andrzej 375
 Stelmach, Paweł 389, 399
 Stolarz, Wojciech 409
 Strusinskiy, Pavel M. 63
 Stupiec, Emil 421
 Sugier, Jarosław 431
 Surmacz, Tomasz 439
 Szlenk, Marcin 449
 Tomala, Radosław 501
 Toporkov, Victor 459
 Toporkova, Anna 459
 Tóth, Tamás 469
 Tselishchev, Alexey 459
 Tylman, Wojciech 479, 489, 501
 Vörös, András 469
 Walker, Martin 141
 Walkowiak, Tomasz 83, 307, 421
 Wąs, Jarosław 285
 Waszyrowski, Tomasz 501
 Wenerski, Maciej 501
 Woda, Marek 409
 Wódczak, Michał 513
 Wojciechowski, Bartosz 439
 Yann-Guirec, Manac'h 523
 Yashina, Marina V. 531
 Yemelyanov, Dmitry 459
 Zalewski, Andrzej 255
 Zieliński, Zbigniew 375
 Zuberek, W.M. 543