

Juan González Nieto
Wolfgang Reif
Guojun Wang
Jadwiga Indulska (Eds.)

LNCS 5586

Autonomic and Trusted Computing

6th International Conference, ATC 2009
Brisbane, Australia, July 2009
Proceedings

 Springer

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Juan González Nieto Wolfgang Reif
Guojun Wang Jadwiga Indulska (Eds.)

Autonomic and Trusted Computing

6th International Conference, ATC 2009
Brisbane, Australia, July 7-9, 2009
Proceedings

Volume Editors

Juan González Nieto
Queensland University of Technology
Information Security Institute
GPO Box 2434, Brisbane, QLD 4001, Australia
E-mail: j.gonzalezniето@qut.edu.au

Wolfgang Reif
University of Augsburg, Institute of Computer Science
Department of Software Engineering and Programming Languages
86135 Augsburg, Germany
E-mail: reif@informatik.uni-augsburg.de

Guojun Wang
Central South University
School of Information Science and Engineering
Changsha, Hunan 410083, China
E-mail: csgjwang@gmail.com

Jadwiga Indulska
The University of Queensland
School of Information Technology and Electrical Engineering
Brisbane, QLD 4072, Australia
E-mail: jaga@itee.uq.edu.au

Library of Congress Control Number: Applied for

CR Subject Classification (1998): D.2, C.2, D.1.3, D.4, E.3, H.4, K.6

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743
ISBN-10 3-642-02703-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-02703-1 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© Springer-Verlag Berlin Heidelberg 2009
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12707553 06/3180 5 4 3 2 1 0

Preface

This volume contains the proceedings of ATC 2009, the 6th International Conference on Autonomic and Trusted Computing: Bringing Safe, Self-x and Organic Computing Systems into Reality. The conference was held in Brisbane, Australia, during July 7–9, 2009. The conference was technically co-sponsored by the IEEE and the IEEE Computer Society Technical Committee on Scalable Computing. ATC 2009 was accompanied by three workshops on a variety of research challenges within the area of autonomic and trusted computing.

ATC 2009 is a successor of the First International Workshop on Trusted and Autonomic Ubiquitous and Embedded Systems (TAUES 2005, Japan), the International Workshop on Trusted and Autonomic Computing Systems (TACS 2006, Austria), the Third International Conference on Autonomic and Trusted Computing (ATC 2006, China), the 4th International Conference on Autonomic and Trusted Computing (ATC 2007, Hong Kong), and the 5th International Conference on Autonomic and Trusted Computing (ATC 2008, Norway)

Computing systems including hardware, software, communication and networks are growing dramatically in both scale and heterogeneity, becoming overly complex. Such complexity is getting even more critical with the ubiquitous permeation of embedded devices and other pervasive systems. To cope with the growing and ubiquitous complexity, autonomic computing focuses on self-manageable computing and communication systems that exhibit self-awareness, self-configuration, self-optimization, self-healing, self-protection and other self-x operations to the maximum extent possible without human intervention or guidance. Organic computing additionally emphasizes natural-analogue concepts like self-organization and controlled emergence. Any autonomic or organic system must be trustworthy to avoid the risk of losing control and to retain confidence that the system will not fail. Trust and/or distrust relationships in the Internet and in pervasive infrastructures are key factors to enable dynamic interaction and cooperation of various users, systems and services. Trusted computing aims at making computing and communication systems as well as services available, predictable, traceable, controllable, assessable, sustainable, dependable, persistent, security/privacy protectable, etc.

The ATC 2009 conference provided a forum for engineers and scientists in academia, industry and government to exchange ideas and experiences in all technical aspects related to autonomic/organic computing and trusted computing. There was a large number of paper submissions (52), representing 17 countries and regions, from Asia, Europe, North America and the Pacific. All submissions were reviewed by at least three members of the Technical Program Committee or external reviewers. In order to allocate as many papers as possible and keep the high quality of the conference, we decided to accept 17 regular papers for presentation, reflecting a 33% acceptance rate. The contributed papers were

supplemented by a keynote address by L. Jean Camp (Indiana University), titled “Design for Trust in Ambient and Ubiquitous Computing.” The proceedings also include an invited paper by Alva Couch, Mark Burgess and Marc Chiarini, titled “Management Without (Detailed) Models.”

Organization of conferences with a large number of submissions requires a lot of hard work and dedication from many people. We would like to take this opportunity to thank numerous individuals whose work made this conference possible and ensured its high quality. We wish to thank the authors of submitted papers, as they contributed to the conference technical program. We wish to express our deepest gratitude to all Program Committee members and external reviewers for their excellent job in the paper review process, as well as the Steering Committee and Advisory Committee for their continuous advice.

We further acknowledge the excellent work of the Workshop Chairs in organizing the three workshops, and the Panel Chairs in organizing the ATC panel. Our thanks also go to the Publicity Chairs for advertising the conference and to the Web Chair for managing the conference website. Last but not least, we would like to thank the NICTA Queensland Lab for conference support and The University of Queensland for hosting the conference.

July 2009

Juan González Nieto
Wolfgang Reif
Guojun Wang
Jadwiga Indulska

Organization

Executive Committee

General Chairs	Jadwiga Indulska, University of Queensland, Australia Hartmut Schmeck, Karlsruhe Institute of Technology, Germany
Program Chairs	Juan González Nieto, Queensland University of Technology, Australia Wolfgang Reif, University of Augsburg, Germany Guojun Wang, Central South University, China
Program Vice Chairs	Dimitri Botvich, Waterford Institute of Technology, Ireland Jurgen Branke, Karlsruhe Institute of Technology, Germany Noria Foukia, Otago University of Otago, New Zealand Seng Loke, LaTrobe University, Australia
Advisory Committee Chairs	Christian Muller-Schloer, University of Hannover, Germany Chin-Chen Chang, Feng Chia University, Taiwan
Steering Committee	Jianhua Ma (Chair), Hosei University, Japan Laurence T. Yang (Chair), St. Francis Xavier University, Canada Hai Jin, Huazhong University of Science and Technology, China Jeffrey J.P. Tsai, University of Illinois at Chicago, USA Theo Ungerer, University of Augsburg, Germany
Workshops Chairs	Willy Susilo, University of Wollongong, Australia Xiaobo Zhou, University of Colorado at Colorado Springs, USA
International Advisory Committee	Jiannong Cao, Hong Kong Polytechnic University, Hong Kong, China Chin-Chen Chang, Feng Chia University, Taiwan Jingde Cheng, Saitama University, Japan Zhong Chen, Peking University, China Petre Dini, Cisco Systems, USA

	Jadwiga Indulska, University of Queensland, Australia
	Victor C.M. Leung, University of British Columbia, Canada
	David Ogle, IBM, USA
	Manish Parashar, Rutgers University, USA
	Franz J. Rammig, University of Paderborn, Germany
	Omer F. Rana, Cardiff University, UK
	Kouichi Sakurai, Kyushu University, Japan
	Hartmut Schmeck, Karlsruhe Institute of Technology, Germany
	Xinmei Wang, Xidian University, China
	Stephen S. Yau, Arizona State University, USA
	Mazin Yousif, Intel, USA
Publicity Chairs	Xiaoyuan Gu, Technical University of Braunschweig, Germany
	Yan Wang, Macquarie University, Australia
	Naixue Xiong, Georgia State University, USA
	Zheng Yan, Nokia Research Center, Finland
	Justin Zhan, Carnegie Mellon CyLab, Japan
International Liaison Chairs	Luo Junzhou, Southeast University, Nanjing, China
	Xiaolin (Andy) Li, Oklahoma State University, USA
	Tai-hoon Kim, Hannam University, Korea
	Roy Sterritt, University of Ulster at Jordanstown, UK
	Sajid Hussain, Acadia University, Canada
Industrial Liaison Chairs	Martin Gilje Jaatun, SINTEF, Norway
	Weidong Kou, IBM, China
Award Chair	Chunming Rong, University of Stavanger, Norway
Web Administration Chair	Peizhao Hu, NICTA, Australia

Program Committee

Hussein Abbass	ADFA, Australia
Lawrie Brown	ADFA, Australia
Kefei Chen	Shanghai Jiaotong University, China
Liquan Chen	HP Labs, UK
Yuanshun Dai	Indiana University-Purdue University, USA
Dengguo Feng	Institute of Software, Chinese Academy of Sciences, China
Sandro Gaycken	University of Stuttgart, Germany
Christopher Gill	Washington University in St. Louis, USA
Andrzej Goscinski	Deakin University, Australia

Jinhua Guo	University of Michigan at Dearborn, USA
Peter Gutman	University of Auckland, New Zealand
Gernot Heiser	University of New South Wales, Australia
Mike Hinchey	Lero, Ireland
Xiaolong Jin	University of Bradford, UK
Audun Jøsang	University of Oslo, Norway
Sy-Yen Kuo	National Taiwan University, Taiwan
Mirosław Kutylowski	Wroclaw University of Technology, Poland
Peter Lindsay	University of Queensland, Australia
Javier Lopez	University of Malaga, Spain
Antonio Maa Gomez	University of Malaga, Spain
Gregorio Martinez	University of Murcia, Spain
Daniel Merkle	University of Southern Denmark, Denmark
Geyong Min	University of Bradford, UK
Chris Mitchell	RHUL, UK
Jan Newmarch	Monash University, Australia
Guenther Pernul	University of Regensburg, Germany
Josef Pieprzyk	Macquarie University, Australia
Fang Qi	Central South University, China
Franz Rammig	University Paderborn, Germany
Jason Reid	Queensland University of Technology, Australia
Jason Smith	Queensland University of Technology, Australia
Miguel Soriano	Technical University of Catalonia, Spain
Ron Steinfeld	Macquarie University, Australia
Jürgen Teich	University of Erlangen-Nürnberg, Germany
Theo Ungerer	University of Augsburg, Germany
Vijay Varadharajan	Macquarie University, Australia
Guilin Wang	University of Birmingham, UK
Huaxiong Wang	Macquarie University, Australia
Hank Wolfe	University of Otago, New Zealand
Dong Xiang	Tsinghua University, China
Yang Xiang	Central Queensland University, Australia
Liudong Xing	University of Massachusetts Dartmouth, USA
Kun Yang	University of Essex, UK
Baoliu Ye	Nanjing University, China
Huanguo Zhang	Wuhan University, China
Jianying Zhou	Institute for Infocomm Research, Singapore
Deqing Zou	Huazhong University of Science and Technology, China

External Reviewers

Shane Balfé	Gang Chen	Sabine Helwig
Michael Brock	Ning Chen	Ihor Kuz
Christian Broser	Ludwig Fuchs	Lei Li

Lei Li
Guanfeng Liu
Guanfeng Liu
Kai Liu
Florian Nafz
Aarthi Nagarajan
Michael Netter

Felix Reimann
Rolf Schillinger
Hella Seebach
Jan-Philipp Steghöfer
Udaya Tupakula
Hua Wang
Guofu Xiang

Lin Yang
Rehana Yasmin
Weiliang Zhao
Daniel Ziener
Tobias Ziermann

Table of Contents

Keynote Speech

Design for Trust in Ambient and Ubiquitous Computing	1
<i>L. Jean Camp</i>	

Organic and Autonomic Computing

Towards an Organic Network Control System	2
<i>Sven Tomforde, Marcel Steffen, Jörg Hähner, and Christian Müller-Schloer</i>	

A Universal Self-organization Mechanism for Role-Based Organic Computing Systems	17
<i>Florian Nafz, Frank Ortmeier, Hella Seebach, Jan-Philipp Steghöfer, and Wolfgang Reif</i>	

Towards Self-organization in Automotive Embedded Systems	32
<i>Gereon Weiss, Marc Zeller, Dirk Eilers, and Rudi Knorr</i>	

Analyzing the Behavior of an Artificial Hormone System for Task Allocation	47
<i>Uwe Brinkschulte and Alexander von Renteln</i>	

A Software Test Cases Automated Generation Algorithm Based on Immune Principles	62
<i>Junmin Ye, Zemei Zhan, Cong Jin, and Qingguo Zhang</i>	

Management without (Detailed) Models	75
<i>Alva L. Couch, Mark Burgess, and Marc Chiarini</i>	

Formal Development of Self-organising Systems	90
<i>Graeme Smith and Jeffrey W. Sanders</i>	

Using Reinforcement Learning for Multi-policy Optimization in Decentralized Autonomic Systems – An Experimental Evaluation	105
<i>Ivana Dusparic and Vinny Cahill</i>	

Trusted Computing

SAConf: Semantic Attestation of Software Configurations	120
<i>Hua Wang, Yao Guo, and Xiangqun Chen</i>	

ALOPA: Authorization Logic for Property Attestation in Trusted
Platforms 134
Aarthi Nagarajan, Vijay Varadharajan, and Michael Hitchens

Wireless Sensor Networks

Employed BPN to Multi-sensors Data Fusion for Environment
Monitoring Services 149
Wen-Tsai Sung

Argus: A Light-Weighted Secure Localization Scheme for Sensor
Networks 164
Wen Tao Zhu and Yang Xiang

Trust

A Methodology towards Usable Trust Management 179
Zheng Yan and Valteri Niemi

Formalizing Trust Based on Usage Behaviors for Mobile
Applications 194
Zheng Yan and Rong Yan

Toward Trustworthy Semantic Web Service Discovery and Selection 209
Jing Li, Dianfu Ma, Jun Han, and Xiang Long

Fuzzy Regression Based Trust Prediction in Service-Oriented
Applications 221
Lei Li, Yan Wang, and Vijay Varadharajan

Theories of Trust for Communication Protocols 236
Ji Ma, Mehmet A. Orgun, and Abdul Sattar

Trust and Reputation Policy-Based Mechanisms for Self-protection in
Autonomic Communications 249
*Martin Serrano, Sven van der Meer, John Strassner,
Stefano De Paoli, Aphra Kerr, and Cristiano Storni*

Author Index 269

Design for Trust in Ambient and Ubiquitous Computing

L. Jean Camp

Indiana University
School of Informatics
ljcamp@indiana.edu

Abstract. Ambient and ubiquitous computing systems are characterized by interfaces so non-traditional that these are often not recognized as computer interactions during use. Further these systems may be always on, embedded nearly invisibly into physical surroundings. Thus empowering individuals to control their own privacy and security in ubiquitous computing is particularly problematic. This is further complicated by the fact that privacy is personal and contextual, while system design is centralized. In this presentation, I describe the designs and associated research methods for developing and evaluating privacy-aware ubicomp for in-home use. The project focused on integrating individuals risk perceptions with respect to security and privacy into the technology, so that choices which appear to be risk-mitigating do indeed mitigate risk. Using six prototypes, the presentation illustrates design for trust in home-based ubicomp.

Towards an Organic Network Control System

Sven Tomforde, Marcel Steffen, Jörg Hähner, and Christian Müller-Schloer

Leibniz Universität Hannover
Institute of Systems Engineering
Appelstr. 4, 30167 Hannover, Germany
{tomforde,steffen,haehner,cms}@sra.uni-hannover.de

Abstract. In recent years communication protocols have shown an increasing complexity – especially by considering the number of variable parameters. As the performance of the communication protocol strongly depends on the configuration of the protocol system, an optimal parameter set is needed to ensure the best possible system behaviour. This choice does not have a static character as the environment changes over time and the influencing factors of the optimisation are varying. Due to this dynamic environment an adaptation depending on the current situation on the particular node within a communication network is needed. This paper introduces an Organic Network Control system which is able to cover this task and it also demonstrates the strengths of the proposed approach by applying the system to a Peer-to-Peer protocol and evaluating the achieved results.

1 Introduction

Due to the continuously increasing interconnectedness and integration of large distributed computer systems and the dramatical growth of communication need new protocols are being developed continuously. Simultaneously, researchers and engineers try to guarantee the sustainability of such systems by optimising and enhancing existing algorithms. This leads to a growing complexity of the particular methods and a rapidly increasing number of possibilities to configure the resulting systems. This complexity of the configuration task leads to the need of new ways to guarantee a good approximation of the optimal behaviour of particular nodes within a network and provide an automatic adaptation to the needs of the system's user. Although the system should be able to cope with different situations, which might not have been foreseen during the development, the user does not want to have additional effort to configure or administrate his system.

Organic Computing (OC - cf. [1]) is a recent research area which focuses on building self-organised systems to solve complex problems. Autonomous entities are acting without strict central control and achieve global goals although their decisions are based on local knowledge. The authors assume that due to the complexity of the particular tasks not all situations can be foreseen during the development process of the system. Therefore, the system must be adaptive and equipped with learning capabilities, which leads to the ability to learn new actions and strategies

for previously unknown situations. The self-control of network entities is also part of the focus of Autonomic Computing (AC - cf. [2]).

Considering the complex problem which arises with the increasing communication demands, OC seems to be a useful approach to build an adaptive network control system. Basic OC techniques like the generic Observer/Controller architecture [3] will be used to enable the demanded properties of the system. Related to another real-world OC example – the control of urban traffic lights as introduced by Prothmann et al. in [4] – our system uses two basic techniques: (1) A Learning Classifier System [5] is used to on-line adapt the network protocol. (2) A Genetic Algorithm [6] in combination with a standard network simulation tool (NS/2, see [7]) is responsible for the creation of new rules.

This paper presents an adaptive and automated system for the dynamic and self-organised control of network protocol parameters (e. g. values for timeouts, maximum number of re-transmissions, number of open connections, etc.). Section 2 introduces evolutionary computing as basic approach as well as recent work done for automatic network protocol parameter optimisation. In Section 3 the architecture and general approach of the proposed system are described combined with details on necessary modifications of the used components (e. g. adaptation of Learning Classifier Systems). Afterwards, Section 4 demonstrates the strengths of our approach by evaluating a BitTorrent-based scenario [8]. The performance of the system is measured off-line by comparing the results of our system with the standard configurations and on-line by demonstrating the increase of performance and usability achieved by the adaptation of the network control system. Finally, Section 5 summarises the approach and names further improvements of the system and next steps of the research.

2 State of the Art

The approach to network control presented by this paper is strongly related to a number of different areas of research. Within this section a brief introduction in the basic concepts needed for the system is given.

2.1 Network Protocol Optimisation

The performance of a network protocol depends highly on the configuration of the parameter set. Therefore, the optimisation of protocol parameter sets is a main task for the development of a new network protocol or the adaptation of an existing one. To determine the best-fitting set of parameters a network engineer could try to manually choose parameters and continue using a directed trial-and-error approach. Alternatively, he could rely on an automated system as the effort for a manual optimisation increases exponentially with the number of parameters and the size of the configuration space per parameter.

The already existing approaches for automated network parameter optimisation are characterised by specific solutions for particular protocols. Although the need of a generic system to fulfill this task has been formulated before (cf. e. g. [9],

p. 14) this does not exist yet. Additionally to the pure optimisation task, another important part of network protocol optimisation is to adapt the parameters of protocols dynamically during the runtime of the application. Both fields of research will be discussed in the remainder of this subsection.

Off-line optimisation of parameter sets deals with the problem to determine a set of parameters for a given protocol that is as close to the optimum as possible. The task is characterised by the needed amount of time and the quality of the solution to be found. As this approach can rely on off-line processing no strict time-restrictions are applied. Additionally, the process can be parallelised as different parameter sets can be processed on different systems. In this context *off-line* means evaluating new possible settings using simulation and thus without interfering with the live-system.

As one example Montana and Redi use an Evolutionary Algorithm (EA) to optimise a full custom communication protocol for military MANETs [10]. A commercial network simulator (OPNET) and real world communication data are used for simulation. The results achieved by the optimisation with the EA are compared to a manual optimisation carried out by the authors themselves. A similar approach for optimising a protocol with an EA is presented by Sözer et al. [11]. They developed a protocol for underwater communications. Again, the optimisation results achieved by an EA are compared to a manual optimisation. In contrast to the Organic Network Control System presented in this paper the solutions are specific systems for the particular protocols, but do not aim at providing a generic system which is adaptable to different protocol types.

Turgut et al. discuss the usage of a Genetic Algorithm to optimise their MANET-based clustering protocol in [12]. This is again a specific implementation for a given protocol. Other techniques can be found in literature, but they all lack the generic character and are not ment to be re-used for other protocols.

On-line adaptation of network controller settings aims at dynamically adjusting the system to a changing environment. Based on the assumption that the environment changes steadily, we define the need to change the parameter set of a network protocol according to the current status of the environment. Therefore, the optimal solution does not depend on a static scenario like for the off-line optimisation problem before, but on the observation of detector values.

In contrast to the off-line part, time restrictions have to be taken into account as the system has to be adapted immediately after observing a changed situation. Additionally, the system has to cover safety- or performance-critical applications where no failures in terms of untested or unknown behaviour is allowed. Both restrictions exclude possible approaches like trial-and-error and long-term learning based on failures. Currently, no standard solution to cover the dynamic on-line adaptation of network protocols has been proposed.

One recent approach by Ye and Kalyanaraman [13] that aims at proposing a possible solution introduces an adaptive random search algorithm, which tries to combine the stochastic advantages of pure random search algorithms with

threshold-based knowledge about extending the search. It is based on their initial system as presented in [14]. It is also applicable to noisy objective functions and is flexible regarding the particular protocol. In contrast to our approach, Ye et al. propose a centralised system that tackles the optimisation task for each node. To allow for such a division of work between a central server and the particular network nodes they have to deal with problems like bandwidth usage, single point of failure, local knowledge accessible from server-side, etc. – which is not necessary when using our solution.

2.2 Evolutionary Computation

Evolutionary Computing (EC - cf. [15]) is a research field in Computer Science. It is designated to study and develop systems using nature-inspired techniques. It aims at providing efficient methods to tackle a large set of optimisation and adaptation problems. EC techniques include Evolutionary Algorithms and Learning Classifier Systems.

Evolutionary Algorithms (EA) are randomised optimisation heuristics that are derived from natural evolution. The process mimics biological evolution and has been applied to several different fields of optimisation problems. Based upon a set (*population*) of randomly generated solutions the evolution-inspired optimisation process evolves new rules by performing genetic operators (e. g. mutation and crossover) by evaluating a certain quality criterion. This criterion is typically called the *objective function* and is maximised – or equivalently minimised during an iterative process. Details on this process and EAs in general are given e. g. by Bäck and Schwefel in [16] or by Mitchell in [6].

Learning Classifier Systems (LCS) have a strong relation to EAs. The concept relies on the goal to learn best fitting solutions. This means that LCS optimise their behaviour depending on currently observed situations and infer on the quality of their choice. They can be applied to problems where for each action some kind of numerical reward can be provided. LCS are based on a rule set, where each rule is representing an information-tuple. This tuple is called *classifier* and contains the three parts *condition*, *action*, and *value* [5].

LCS select a particular classifier based on a given stimulus (e. g. the currently observed situation). The process can be divided in two main steps: selecting a set of matching classifiers and choosing the desired action. The selection task is done by building a *match set* which is a subset of the rule set and contains all classifiers whose condition part matches the current stimulus. These selected classifiers may suggest different actions, which leads to the second step where one action has to be chosen. Therefore, the average values of all distinct actions contained in the match set are calculated by taking into account all classifier values proposing the specific action and being part of the match set. The resulting list of actions is ordered descendingly by their calculated value, and the one with the highest value is applied to the System under Observation and Control (SuOC). All those classifiers proposing the selected action form the *action set*. Afterwards, the reward received from the environment is used to update the values of all classifiers contained in the action set.

Besides the process of selecting actions from a given set of classifiers, the creation and modification of classifiers is of basic interest for the system. To fulfill this task two different approaches are applied: If no matching classifier was found during the building process of the match set, a classifier consisting of a condition matching the current stimulus, a random action, and a default value is added to the rule base. This process is called *covering*. Additionally, sporadically a reproduction cycle is started. Within this cycle some classifiers are chosen to be "parent" individuals, and the genetic operators *crossover* and/or *mutation* are applied to copies of these parents to form offspring which are inserted to the rule base [17].

The concept described above is the basic approach of LCS. A large set of different implementations have been proposed, but most of them rely on the research done by Wilson in e. g. [5,17]. For further details on this topic the reader is referred to Wilson.

3 An Organic Network Control System

This section explains the architecture of our proposed Organic Network Control (ONC) System. Based upon a given network protocol client – the *System under Observation and Control* (SuOC) in terms of Organic Computing – an additional Observer/Controller (O/C) component is used to adapt the parameter set of the SuOC depending on the currently observed situation within the network. To allow for the creation of tested new rules for the Learning Classifier System situated at Layer 1, a simulation-based second layer is added upon the O/C part. The general approach of the architecture as pictured in Fig. 1 is based on the generic O/C architecture presented in [3].

The approach of our architecture is based on one of the key properties of OC – an additional higher layer has to provide additional functionality, but a removal must not affect the operational capabilities of the basic system. We aim at proposing a generic solution, which means that the SuOC itself is exchangeable. Only one restriction on the exchangeability is applied as the system can only cover parametrisable protocols, on which the performance can be measured locally. In contrast to that our system can currently not handle optimisations, where the logic of the protocol has to be adapted as it is based upon a generic character and a black-box approach. The target system provides capabilities to rapidly adapt to changing environments. It also has the typical OC properties of being self-configuring, self-learning, self-optimising, and self-organising.

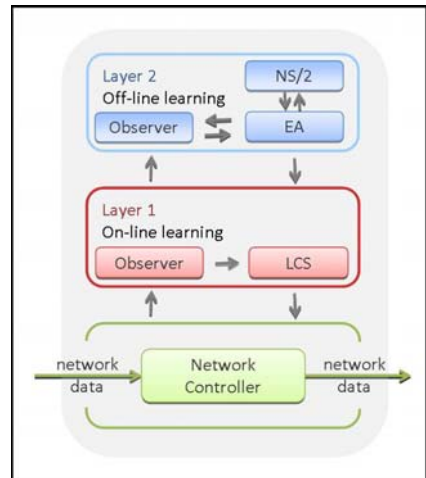


Fig. 1. Architecture for the Organic Network Control system

3.1 Architecture of the ONC System

The ONC system is based on the architecture as depicted in Fig. 1. It is also strongly influenced by the implementation of the generic O/C architecture as introduced by Prothmann et al. in [4] for urban traffic control systems. Therefore, the basic approach to allocate tasks to particular layers and the techniques (e. g. Evolutionary Algorithms and Learning Classifier Systems) have been adopted. But, compared to the urban traffic control system, a range of necessary adjustments had to be applied. Within the remainder of this Section we will describe the three different layers with their components and tasks, followed by the necessary adaptations of the particular technical approaches to realise our system.

Layer 0: System under Observation and Control. The SuOC (see Fig. 2) is a parametrisable Network Controller. Due to the generic concept of the proposed system it is not restricted to a particular set of protocols – the only restriction applied is that it has to provide a set of variable parameters and a local quality criterion (e. g. the duration of a download in Peer-to-Peer systems or a weighted trade-off between the energy consumption and the broadcast-covering for MANETs) for the performance measurement. This means, protocols for media access and applications (e. g. Peer-to-Peer systems) can be controlled in the same way as e. g. wire-based protocols or mobile ad hoc networks (MANETs). A good setup of the variable parameters that match the current condition at the network node has an important influence on the resulting performance for these systems. In the architecture, the parameter setup is optimised on-line by the O/C component in Layer 1.

Layer 1: Observer/Controller for on-line adaptation.

The variable parameters of the SuOC are subject to control interactions of the component situated at Layer 1. As depicted in Fig. 2 this component can be divided into two different parts: an Observer and a Controller. The Observer is responsible for monitoring the situation at the particular node. The description of the situation is aggregated by transforming the measured values to a more abstract and normalised view of the current status. Depending on these figures the second part of the layer – the Controller unit – decides whether an adaptation of the SuOC is needed. This part is realised by a Learning Classifier System (LCS) which maps the input from the Observer to a rule base of possible actions. If the LCS does not contain a matching rule, the rule generation mechanism situated in Layer 2 is triggered in addition to the usage of a covering mechanism.

Layer 2: Off-line optimisation. The existing set of classifiers of the LCS is extended for unforeseen or currently unknown situations by using the rule-creation

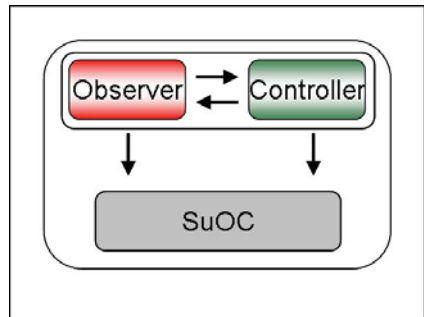


Fig. 2. The SuOC

mechanism of Layer 2. In this context off-line means that it works upon demand of the on-line system at Layer 1 and delivers the new simulation-based created rules as soon as they are generated. Within this layer an EA evolves a new parameter set for the SuOC designated for a given situation. In combination with a widely used standard network simulation tool the EA performs an off-line optimisation. Additionally, a Layer 2-Observer serves as monitoring component to allow for the situation-dependent usage of system resources and queuing of the optimisation tasks of Layer 1.

3.2 Technical Realisation of the On-Line Learning System (Layer 1)

The on-line learning system is situated on Layer 1 of our architecture and consists of two main parts: an Observer and a Controller component.

The Observer is needed to monitor the situation at the node. It measures those figures having influence on the selection of appropriate parameters for the control strategy. This selection depends on the specific controlled protocol and typically contains figures like buffer sizes, delay times, etc. Additionally, protocol-specific parameters like e. g. number of nodes in sending distance for MANET protocols or available system resources like CPU, upload-bandwidth, download-bandwidth, etc. for P2P protocols can be taken into account. The Observer monitors the particular values and aggregates them to an abstract situation description realised as an n -dimensional vector with n equal to the number of observed values. Which figures are used to describe the situation depends on the protocol as e. g. MANETs have other influencing factors than P2P systems.

The Controller contains a LCS, which is based on Wilson's XCS as introduced in [17]. Our modifications of the original implementation lead to a real-valued solution which receives an n -dimensional vector from the Layer 1-Observer describing the current situation. For the composition of the classifier this results in a condition part containing n interval predicates forming an n -dimensional hyper-rectangle where all states of the observed influencing factors are encoded. The action part of the classifier is a parameter set for the control of the network node. The selection process is done analogously to the method proposed by Wilson (further details can be found e. g. in [17]): In a first step the LCS selects all classifiers matching the current situation description. Afterwards, that action is chosen from the created match set which is predicted to be most appropriate for the current situation. The third step applies these parameters to the SuOC. Finally, the LCS receives a reward for its selection and updates the evaluation of all classifiers contained in the action set. To determine the reward, the LCS compares the current situation description with the prediction (e. g. measured as amount of data downloaded in a given time interval) made by the classifier when building the action set.

Nevertheless no matching classifier might be found (which means the match set is empty). In this case an action to be applied to the SuOC is needed. But, the standard randomised approach introduced by Wilson can not be used as a real-world system depends on useful actions. We assume that a classifier whose

condition part is located next to the current situation description – although it does not match it – is better than any other one existing within the rule set. Due to this assumption a covering process is executed which selects the "nearest" classifier in terms of the Euclidian Distance calculated for the n dimensional vectors and using the centroids of the intervals used for each interval predicate. This classifier is copied, its condition part is adapted to the current situation description (using a standard interval size around the given situation), and it is added to the rule set. Based on this simple process we ensure to only use tested actions and we also ensure that at least one rule is contained in the match set.

3.3 Technical Realisation of the Off-Line System (Layer 2)

The on-line adaptation of the system works depending on the *existing* set of classifiers. If no classifier matches the input a new one is generated using the covering mechanism as described before. This might not lead to a sufficiently well performing solution – particularly if a classifier is selected, which is located far from the current condition within the n dimensional space. Due to this inconsistency a new classifier is needed. In Wilson's XCS, this is realised using two varying techniques: crossover and a random kind of evolution by creating *childs* from particular *parents*. This randomised approach to creating new classifiers is infeasible for most of the real-world problems and especially for the network control system. The largest part of the created classifiers would lead to an unacceptable performance or affect the usage of the underlying SuOC. Therefore, this task is delegated to the highest layer of our architecture, where a simulation-based approach is used to create new and tested classifiers.

The Observer is responsible for capturing the current situation description provided by the Observer on Layer 1. As the current usage of system components (in terms of CPU, RAM, etc.) might have influence on the selection process of the LCS and being part of the condition of the classifier it might be also restricted for Layer 2. Therefore, this Layer 2-Observer is needed to receive the status of the observed variables from the Layer 1-Observer. It also manages the queue of optimisation tasks as the simulation-based creation of new rules needs considerable effort in terms of system resources.

The Evolutionary Algorithm is responsible for evolving new classifiers. Based on the procedure described in Section 2.1 a new action is generated for the observed situation. Therefore, the EA is used with the configuration listed in Table [1](#). The number of generations is equal to the number of processed iterations until the EA stops – by trend, a higher number would lead to a higher quality, but also more time is spent. A similar statement can be formulated for the population size. By defining the number of children the fraction of new individuals per cycle is defined. Finally, the mutation and crossover rate have significant influence on the modification of new individuals compared to the existing individuals contained in the previous population.

Table 1. Configuration of the EA

<i>Variable</i>	<i>Value</i>
Number of generations	18 cycles
Population size	10 individuals
Number of children per generation	6 individuals
Mutation rate	0.8 per child
Crossover rate	0.83 per child

The Simulator component to test the parameter sets generated by the EA is realised using NS/2 [7]. This discrete event simulator is a well-known standard solution and has also been used by many authors of network protocols to evaluate their systems. It relies on providing an implementation of the specific protocol as well as a scenario defining the variables to be optimised, the situation of the network, and the environment. The situation depends on the current observation – which means that the system on Layer 2 adapts the NS/2-scenario to the aggregated situation-description as provided by the Observer on Layer 1.

4 Evaluation of the System

This section aims at demonstrating the performance of the ONC system. Initially, we introduced a stand-alone off-line optimisation system for network protocol parameters and evaluated the performance for MANETs and Smart Camera communication protocols in [18]. Based upon this system we extended the architecture to an adaptive and dynamic on-line network control system. As both parts – the on-line and the off-line optimisation – are basic components we evaluated both. Details are given in the remainder of this section. The performance of the off-line optimisation can be measured compared to the usage of the standard parameter set. The evaluation of the on-line part is motivated by the assumption that varying user behaviour – and combined with that the different requirements of the particular situation – can not be foreseen completely at development time. The need of an on-line adaptation is demonstrated by using a prototypical course of day of one person using his/her computer and demonstrating the achieved increase of performance in terms of amount of downloaded data. Therefore, we start with the experimental setup, followed by the description of our underlying scenario, and the used protocol. Finally, we present the results of the evaluation process.

4.1 Experimental Setup

The experimental setup consists of different parts. One basic part is the configuration of the scenarios used by the NS/2 simulator as this has influence on the achievement of the results. Secondly, the evaluation of the on-line part depends on a scenario modelling the course of day of computer-usage. Accompanied by this scenario, the protocol being used is of high interest for the analysis of the results. Therefore, we give a short introduction to the BitTorrent protocol and describe its

variable parameters underlying the optimisation and control process of our system. The evaluation has been performed on a standard PC (CPU: AMD Athlon 3200+, 2.00 GHz and RAM: 1.00GByte).

NS/2 Configuration. The discrete event simulator NS/2 [7] is a well-known standard solution for simulating network behaviour in research. It simulates a given network protocol based on the definition of a scenario. Within this scenario it is specified, how many nodes are taking part in the simulation, which files are transferred, and which events appear during the simulation process. As most important figures to reproduce the simulation and the received results we used the following configuration: number of nodes: 100, number of seeds (peers initially providing the data): 3, size of files: 500MB, and number of files: 1, if no other configuration is mentioned. The maximum possible link bandwidth has been set to 400 KByte/sec for downstream and 40 KByte/sec for upstream.

Scenario. The on-line evaluation of the ONC system is based on a simple scenario, which is inspired by the role model of a student in Computer Science. Thereby, we consider the usage of a standard PC during the course of a day. The user fulfills tasks, which leads to a certain usage of resources. Additionally, he has to sporadically look up information on the Internet and download data, which leads to a demand of up- and download-capacity. Simultaneously, he runs a Peer-to-Peer (P2P) Client to download a high amount of data (e. g. video recordings of lectures). The target of the download is to receive the data files as fast as possible using the P2P-Client without decreasing the usability of the PC for the normal tasks. This means, the P2P Client can only use those system resources not being occupied by the user. The model of the system during the day is depicted in Fig. 3 and currently contains the factors upload-bandwidth and download-bandwidth. This means for the evaluation of this paper, factors like CPU- or RAM-usage remain unconsidered.

Protocol: BitTorrent. The P2P-Client is based on the BitTorrent protocol [8], which is a very popular example for those protocols. Recently, BitTorrent networks

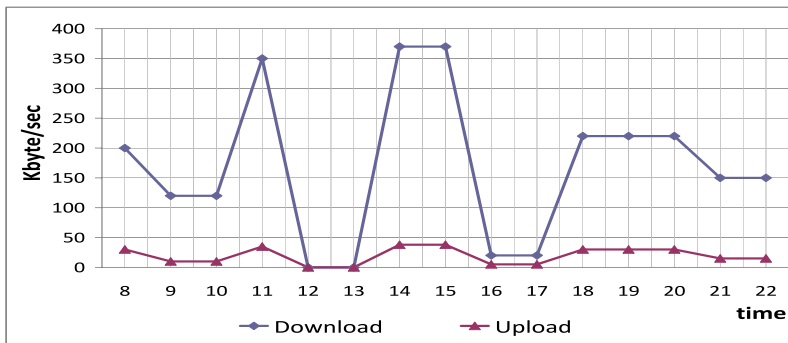


Fig. 3. Usage profile

were responsible for up to 30% of all traffic on the Internet (in the year 2005 – cf. [19]). As standard P2P protocols have to cover several problems like fairness, efficiency, etc., BitTorrent introduced a simple concept to deal with this. Built upon two basic approaches to increase the efficiency and to reduce free-riding – which were both the main problems of previous P2P systems – the system relies on the usage of fragmented files (splitting into sub-files) called *chunks*. Finishing a download of one chunk can immediately allow for the upload of the chunk to other peers. Additionally, each peer controls to whom it uploads data. For further details of the protocol the reader is referred to Cohen’s introduction of BitTorrent in [8].

For the evaluation of our system we use the implementation of a BitTorrent-like protocol for NS/2 presented by Eger et al. in [19] and available on the Internet at [20]. The protocol is called “BitTorrent-like” as it does not implement a specific version of BitTorrent and relies on some simplifications. Although this leads to a decreased functionality of the whole protocol, it still behaves like the standard protocol as most abstractions being made are concerned with the distributed usage within the Internet. For details on the simplifications and the implementation please see [20].

Variable Parameters of the protocol. The BitTorrent Client offers seven different parameters, which can be adapted by the user. They are listed with a short description and their standard configuration in Table 2. Additionally, there are several parameters, which are used in the implementation and might be subject to the optimisation process. Currently we just considered the adaptation of the listed standard parameters – the direct adaptation of the protocol implementation within the simulator is subject to further investigations.

Table 2. Variable parameters of the BitTorrent protocol

<i>Variable</i>	<i>Description</i>	<i>Standard value</i>
NumberOfUnchokes	Number of unchoked connections	4 conn.
ChokingInterval	Interval for unchoking process	10 sec.
RequestPipe	Number of simultaneously requests	5 req.
NumberPeersPerTracker	Number of requested peers	50 peers
MinPeers	Min. number of peers for not re-requesting	20 peers
MaxInitiate	Maximum number of peers for initialisation	40 peers
MaxConnections	Maximum number of open connections	1000 conn.

4.2 Evaluation of the Off-Line Parameter Optimisation

The off-line optimisation is based on the BitTorrent-based NS/2 simulation as presented in Section 4.1. The EA has been configured using the values as listed in Table 2. The resulting values of the simulation are averaged values for at least three simulations, but there are nearly no differences between the runs. For the optimisation the settings of all peers have been adapted to the same values, which means there is no individual configuration. The classifiers are structured according to the

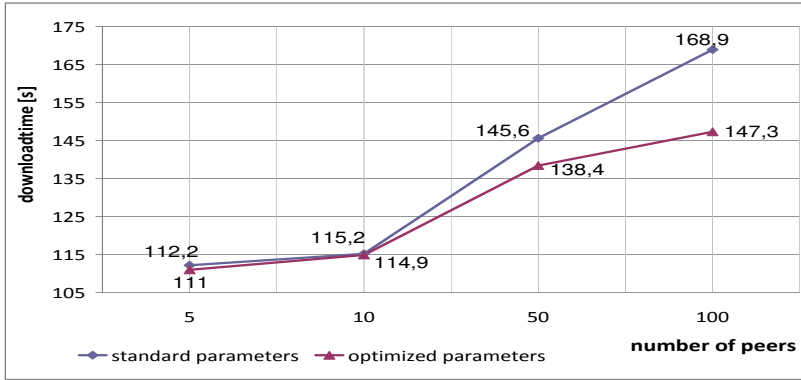


Fig. 4. Optimisation performance with varying number of peers

process described in Section 2.2 and 3.2. The fitness function has been defined as minimising the download time for the particular node.

We investigated the influence of the optimisation for different configurations of the scenario, varying the number of peers and the size of the downloaded files. Fig. 4 shows the results for varying the number of peers (5, 10, 50, and 100 peers) and compares the time needed for the download. For a small number of peers we observed only a small decrease in download time (5 peers: 111.0sec optimised vs. 112.2sec using the standard configuration – increase in performance of 1.1%). For a higher number of peers (100 peers: 147.3sec optimised vs. 168.9sec using the standard configuration – decrease in download time of 14.67%) the performance increases. This means, the number of peers has a significant influence on the possible optimisation performance. In contrast to the number of peers, the file-size to be downloaded seems to have nearly no influence (Optimisation results are between 0.1% and 1.0% better than the standard values).

As one example for an optimised solution, the system evolved the following parameter set (100 peers): NumberOfUnchokes - 3; ChokingInterval - 20; RequestPipe - 8; NumberPeersPerTracker - 24; MinPeers - 20; MaxInitiate - 40; MaxConnections - 1104. The unchanged values for the number of peers can be explained by our scenario as the simulation covers only 100 peers. The adapted values of the choking process can be explained by the fact that our scenario is based on nearly constant connections, which leads to a rarely usage of the un-choking process.

Finally we can state, that the optimised parameter sets lead to an improvement in all tested situations compared to the standard configuration. The evolved configuration achieved an increase in performance of 0.1% to 30.0% (in specific configurations). The time needed for the generation of a new parameter set depends highly on the particular scenario configuration: scenarios with only a few peers and small file-sizes (e. g. 5 peers and 5MB files) needed only a couple of minutes while complex scenarios (100 peers and 1000MB files) needed up to 24 hours per simulation run.

4.3 Evaluation of the Dynamic On-Line Learning System

As demonstrated within the previous subsection, the off-line optimisation leads to a significant increase in performance due to the adaptation of the parameter set. In contrast to the off-line part the on-line system has to adapt directly to its changing environment (the system checks the need for control actions every 3 minutes). Therefore, the system can not wait for the off-line part to be finished as this may take up to 24 hours. The optimisation tasks are queued and processed in the order they arrive.

The evaluation of the on-line system is done by comparing the achieved download rate during the course of day on three consecutive days (see Fig. 5). The usage profile is the same for all days, which means that exactly the same situations are observed by the ONC system. The system starts with a completely empty rule base and therefore initially uses the standard parameter set. Due to the duration of the Layer 2 process the rule base increases slowly. Until the system has the matching parameter set for each situation it uses the already existing rules, a nearby rule (chosen by the covering mechanism), or – for a *completely* unknown situation the standard parameter set.

As neither a matching nor a covering parameter set is available on Day 1 the averaged download rate (165.5 KByte/sec) is equal to the usage of the standard parameter set. Day 2 shows an improved situation-dependent selection of parameter sets (see Fig. 5) which leads to an increased download performance (177.4 KByte/sec – increase of 7.2% compared to the standard configuration). This can be explained by the fact that the ONC system recognises the situation for which a new parameter set has been evolved and chooses this (e. g. first situation from 8 to 9 o'clock). Additionally, the covering mechanism has more options and is able to use close-by parameter sets. Finally, on Day 3 the system chooses always the optimum parameter set (average download rate of 199.3 KByte/sec – increase of 20.4% compared to the usage of the standard parameter set). For this example, there is no further improvement after Day 3, as this is the optimum. Therefore, the EA has performed 14 runs (one for each new situation).

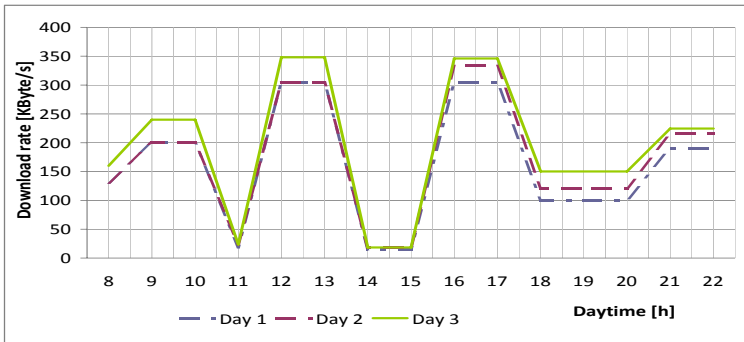


Fig. 5. On-line performance of the ONC system

The covering mechanism is responsible for the increase in performance for situation where currently no optimised rule is available. An example for the quality of this is the time interval from 4pm to 5pm: At day 1 the system uses the standard configuration (304.7 KByte/sec). On day two the covering chooses a close-by rule which leads to an improvement of 9.4% (333.3 KByte/sec). On the third day the optimised value is available (346.1 KByte/sec) leading to an improvement of 13.6% compared to the standard parameter set.

5 Summary and Outlook

This paper presented a system for the dynamic adaptation of network protocol parameters based on the observation of the current situation at the particular node of a communication network. It allows for the dynamical change of the configuration of the underlying protocol to ensure the best possible performance at each time. We discussed the architecture of the system and explained the segmentation into an on-line learning component and an off-line rule-generation mechanism with the particularly used techniques.

We also showed the portability of the generic O/C architecture and its two-layered special variant as presented by Prothmann et al. in [4]. Furthermore, the performance of the system has been demonstrated for a P2P-based scenario. Thereby, the situation-based generation of optimised parameter sets has been shown and compared to the performance of a system with the standard configuration. The results of this off-line optimisation are promising. Afterwards, we demonstrated the performance of the on-line system based on the assumed system usage during the course of day. Again, the system outperformed the standard configuration. We also discussed the influence of factors like number of nodes in the neighbourhood or file-sizes.

For the on-line system we showed the increase in performance over time. Based on an empty rule set the system autonomously learns new rules depending on the recently observed unknown situation. It creates new, optimised, and tested rules and is able to re-use this new rules if the situation occurs again. This leads to a increased performance of the system in terms of the objective function.

Currently we are working on the adaptation of the presented system to different protocols. Therefore, we aim at enabling the system to work on MANET nodes in a first step. Afterwards, cable-based protocols are focused up to optimising TCP/IP. The second main task is to extend the scope from one protocol to several at once – where again the TCP/IP protocol stack seems to be a good target system. Although we aim at an adaptation of the parameter sets, the TCP friendliness should be guaranteed.

Finally, the existing system will be used to investigate collaboration aspects. As we deal with several similar nodes within a homogeneous communication network, the learning and adaptation speed might be increased by collaboration between neighbouring nodes. The focus will be on enabling the network control systems to exchange rules, agree on the distribution of optimisation tasks, or evolving a network-wide optimisation.

References

1. Schmeck, H.: Organic Computing – A new vision for distributed embedded systems. In: Proceedings of the 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2005), pp. 201–203 (2005)
2. Kephart, J.O., Chess, D.M.: The Vision of Autonomic Computing. *IEEE Computer* 36(1), 41–50 (2003)
3. Branke, J., Mnif, M., Müller-Schloer, C., Prothmann, H., Richter, U., Rochner, F., Schmeck, H.: Organic Computing – Addressing complexity by controlled self-organization. In: Proc. of the 2nd Intern. Symp. on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2006), pp. 185–191 (2006)
4. Prothmann, H., Rochner, F., Tomforde, S., Branke, J., Müller-Schloer, C., Schmeck, H.: Organic control of traffic lights. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) ATC 2008. LNCS, vol. 5060, pp. 219–233. Springer, Heidelberg (2008)
5. Wilson, S.W.: ZCS: A zeroth level classifier system. *Evolutionary Computation* 2(1), 1–18 (1994)
6. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1996)
7. Web: The Network Simulator - NS/2, <http://www.isi.edu/nsnam/ns/>
8. Cohen, B.: Incentives Build Robustness in BitTorrent. In: Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley (2003)
9. Kunz, T.: Multicasting in mobile ad-hoc networks: achieving high packet delivery ratios. In: CASCON 2003: Proc. of the 2003 conference of the Centre for Advanced Studies on Collaborative research, Toronto, Canada, pp. 156–170. IBM Press (2003)
10. Montana, D., Redi, J.: Optimizing parameters of a mobile ad hoc network protocol with a genetic algorithm. In: GECCO 2005: Proc. of the 2005 conference on Genetic and evolutionary computation, pp. 1993–1998. ACM, New York (2005)
11. Sözer, E.M., Stojanovic, M., Proakis, J.G.: Initialization and routing optimization for ad-hoc underwater acoustic networks. In: Proceedings of Opnetwork 2000 (2000)
12. Turgut, D., Daz, S., Elmasri, R., Turgut, B.: Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach. In: Proc. of the IEEE Global Telecommunications Conference (GLOBECOM 2002), pp. 62–66 (2002)
13. Ye, T., Kalyanaraman, S.: An adaptive random search algorithm for optimizing network protocol parameters. Technical report, Rensselaer Polytechnic Inst. (2001)
14. Ye, T., Harrison, D., Mo, B., Sikdar, B., Kaur, H.T., Kalyanaraman, S., Szymanski, B., Vastola, K.: Network Management and Control Using Collaborative On-line Simulation. In: Proceedings of IEEE ICC, Helsinki, Finland. IEEE, Los Alamitos (2001)
15. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Natural Computing Series, vol. 2. Springer, Berlin (2003)
16. Bäck, T., Schwefel, H.P.: Evolutionary computing: An overview. In: Proceedings of IEEE Conference of Evolutionary Computing (1996)
17. Wilson, S.W.: Classifier fitness based on accuracy. *Evolutionary Computation* 3(2), 149–175 (1995)
18. Tomforde, S., Brammer, F., Hoffmann, M., Hähner, J.: POWEA: A system for automated network protocol parameter optimisation using evolutionary algorithms (2009) (submitted for publication)
19. Eger, K., Hofeld, T., Binzenhöfer, A., Kunzmann, G.: Efficient simulation of large-scale p2p networks: Packet-level vs. flow-level simulations. In: 2nd Workshop on the Use of P2P, GRID and Agents for the Development of Content Networks (UPGRADE-CN 2007), Monterey Bay, USA, pp. 9–16 (2007)
20. Eger, K.: Simulation of BitTorrent Peer-to-Peer Networks in ns-2, <http://www.tu-harburg.de/et6/research/bittorrentsim/index.html>

A Universal Self-Organization Mechanism for Role-Based Organic Computing Systems*

Florian Nafz, Frank Ortmeier, Hella Seebach, Jan-Philipp Steghöfer,
and Wolfgang Reif

Lehrstuhl für Softwaretechnik und Programmiersprachen,
Universität Augsburg, D-86135 Augsburg
{nafz,ortmeier,seebach,steghoefer,reif}@informatik.uni-augsburg.de

Abstract. An Organic Computing system has the ability to autonomously (re-)organize and adapt itself. Such a system exhibits so called self-x properties (e.g. self-healing) and is therefore more dependable as e.g. some failures can be compensated. Furthermore, it is easier to maintain as it automatically configures itself and more convenient to use because of its automatic adaptation to new situations. On the other hand, design and construction of Organic Computing systems is a challenging task. The Organic Design Pattern (ODP) is a design guideline to aid engineers in this task.

This paper describes a universal reconfiguration mechanism for role-based Organic Computing systems. If a system is modeled in accordance with the ODP guideline, reconfiguration can be implemented generically on the basis of an of-the-shelf constraint solver. The paper shows how Kodkod can be used for this and illustrates the approach on an example from production automation.

1 Introduction

Increasing complexity and steadily rising requirements more and more often become dominant problems during system development and maintenance. Organic Computing (OC) [9] and Autonomic Computing (AC) [6] are trying to tackle these challenging aspects. The basic idea is to build systems, such that they can autonomously adapt to changing requirements, optimize themselves at runtime for better performance or compensate failures by smart counter measures. These abilities are often referred to as self-adapting, self-optimizing and self-healing.

Although such self-x properties are highly desirable, it is still a challenging task to design and construct systems with such abilities. The Organic Design Pattern (ODP) is a design guideline for a specific-class of self-x systems. It provides efficient support for design and specification of an Organic Computing system. However it is still a challenging task to refine this design to an actual implementation. This paper shows how the problem of implementing a self-reconfiguration

* This research is partly sponsored by the priority program “Organic Computing” (SPP OC 1183) of the German research foundation (DFG).

algorithm according to a given specification can be solved in a generic way. This is very valuable during system construction as there exist many, very elaborate construction processes for the functional parts of an OC system while implementing the *organic* behavior is often a very problem-specific and creative task. Technically, this is achieved by translating the corresponding design artifacts and OCL constraints which describe the behavioral and structural properties of a system into a model for a generic constraint solver (in this case: Kodkod).

The paper is organized as follows: Section 2 gives a brief overview of the Organic Design Pattern and the system class this paper focuses on. In Section 3 an introduction to Kodkod and the translation of necessary ODP artifacts into Kodkod’s (relational) modeling language are given. An illustration on a real world example from production automation is shown in Section 4. Finally, Section 5 concludes the paper with some related approaches and a brief outlook on future work.

2 Organic Design Pattern

The Organic Design Pattern (ODP) [12] is a design principle for a broad class of self-x systems, namely those which consist of a set of independent components interacting with each other and where reconfiguration and adaptation respectively can be expressed as a reallocation of roles. The components of such a system have to provide redundancy with regard to their functionality to enable such reallocations at run time. This means the components must have several capabilities they can use to fulfill different roles. The computation of a correct new role allocation then takes into account the possible interactions between the components, the capabilities of the components and the task that has to be achieved by the whole system.

The systems regarded in this paper are distinguished by changing tasks during runtime and the possibility to process several resources with different tasks at the same time. Examples for such systems are sensor networks, distributed smart devices which provide context sensitive services, or adaptive production automation systems. Furthermore, the systems can run in a degraded mode in which a task is only partially fulfilled, thus compensating for failure as long as possible. The core of the pattern which allows for modeling such systems is an elaborate role concept. The model has been based on a precise semantics which allows to define and measure self-x properties [3]. Additionally, the reconfiguration process can be described on an abstract level very intuitively.

The following paragraphs give a very brief introduction to important concepts of ODP and the “Restore-Invariant-Approach” [4] for specification of reconfiguration algorithms. An application of this design concepts to a real-world case study will be shown in Section 4.

2.1 Static View

An ODP system consists of *Agents* which process *Resources* with one or more of the agents’ *Capabilities* according to a given *Task*. A *Task* describes how a given

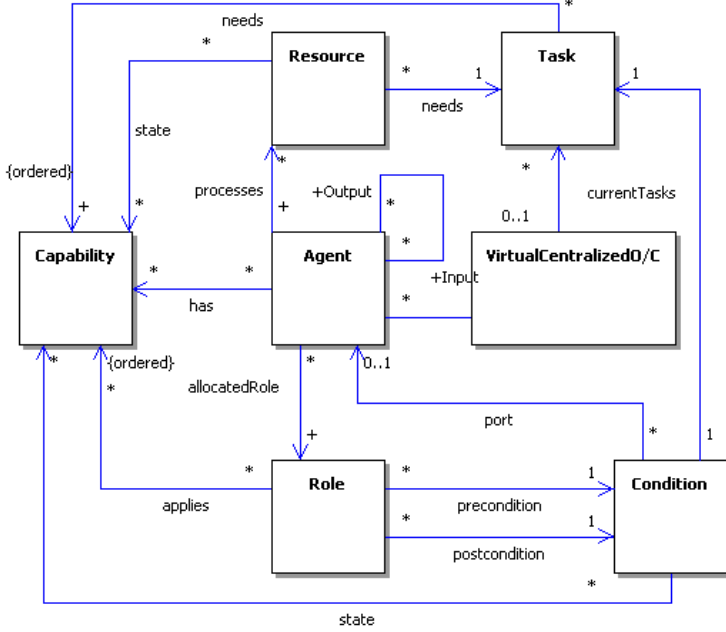


Fig. 1. Organic Design Pattern

Resource should be processed. It is a sequence of *Capabilities* which should be applied to the *Resource*. The static view of such a system is shown in Figure 1.

Every *Agent* is characterized by the *Capabilities* it can provide and the agents to which/from which it can give/receive *Resources*. Which *Capability* an *Agent* performs in a specific situation is determined by its *Role*. An *Agent* can have several *Roles*. The association *allocatedRole* represents the mapping of *Agents* to *Roles* which will be called *role allocation* in the rest of the paper.

Self-organization in this class of systems is described as a role allocation problem. A *Role* is a 3-tuple (*precondition*, *Capabilities*, *postcondition*) of a precondition, a sequence of *Capabilities* that need to be applied and a postcondition. The precondition describes which resources are accepted by the agent and which other agent provides them (*port*). The postcondition describes how the resource is labeled and which agent should receive it. *Conditions* are 3-tuples of a target *Agent* from which, respectively to which, the *Resource* is taken, respectively given, the current *state* of the *Resource* and the *Task* that needs to be done. The reconfiguration mechanism is modeled in the concept of the *VirtualCentralizedO/C* (Observer/Controller [11], O/C). This component encapsulates the reconfiguration algorithm on design level. The output of the reconfiguration algorithm is a new allocation of *Roles* to *Agents* that restores the system to a state in which it is able to fulfill its tasks again.

2.2 Dynamic View

The dynamics of an ODP system are relatively simple. All *Agents* run asynchronously parallel. Dynamics can be split into two sub-domains: behavior during ‘normal’ (i.e. productive) phases and behavior when self-organization occurs.

During normal operation, interaction between *Agents* is done by passing *Resources* between *Agents*. Whenever an *Agent* receives a *Resource*, it chooses one of its *allocatedRoles* according to the *precondition* and the *Capabilities* that have to be performed. Then the *Agent* applies the *Capabilities* defined in this *Role* to the *Resource*, refreshes the state and task of the *Resource* according to these *Capabilities* and gives the *Resource* to the *Agent* in the *postcondition* of the *Role*¹. Reconfiguration is always triggered when a given role allocation is no longer correct. Then the self-organization phase begins and a new role allocation must be calculated. The fact that the actual role allocation is no longer valid can typically be noticed during runtime by the agents. An example is that an agent loses a capability (maybe because of a hardware failure). The agent will then eventually receive a resource which cannot be processed with the remaining capabilities. This will trigger a reconfiguration (in this context, it is assumed that an agent can detect the loss of a capability). Whenever a reconfiguration is triggered, (1) all agents are informed to stop productive operation, (2) the resources are cleared from the system, (3) data is gathered from the agents and (4) a new, valid role allocation is calculated and distributed. Finally, the system again enters a productive state.

2.3 Specification of Self-x Behavior

One of the major challenges in designing an ODP-based Organic Computing system is specification of the reconfiguration algorithm. In the context of this paper, we restrict ourselves to sequential role execution where each agent performs only one role simultaneously.

The big advantage of using the Organic Design Pattern is that it allows to systematically *design* self-x behavior by specifying properties for role allocations. The properties can be described as OCL constraints [10]. The properties can typically be split into two groups. The first group addresses consistency issues. Some examples are:

```

con1 inv: self.Input -> includesAll(self.
    allocatedRole.precondition.port)
con2 inv: self.Output -> includesAll(self.
    allocatedRole.postcondition.port)
con3 inv: self.allocatedRole.precondition.port
    -> forAll(b:Agent | b.allocatedRole.
    postcondition.port -> includes(self))

```

¹ Note, that the description above implies that the selection of a role generally depends on the *Resource* the *Agent* is actually processing. So there must also be a selection algorithm implemented, which chooses a *Role* if several *allocatedRoles* are applicable at the same time.


```

con4 inv: self.allocatedRole.postcondition.port
  -> forAll(b:Agent | b.allocatedRole.
precondition.port -> includes(self))

```

These four constraints are invariants for the class *Agent*, which means they are defined in the **context** *Agent*. Constraints 1 and 2 assert that an *Agent* can only receive resources from and give resources to *Agents* corresponding to its input and output associations and thus state that roles allocated to an *Agent* have to be consistent with its input and output relations. Constraints 3 and 4 assert that if *Agent* A (referred to as “self”) has *Agent* B as a port in the precondition of one of its *allocatedRoles*, then *Agent* B must have a role where *Agent* A is the port in the postcondition and vice versa. These kinds of constraints have to be considered when a new role-allocation is calculated. They determine the valid configurations for an ODP system and therefore the admissible results of a reconfiguration algorithm.

The second group of constraints addresses properties, which must be monitored during runtime. They can very often (but not always) be decentralized and monitored by individual agents. The most important example is:

```

mon1 inv: self.has -> includesAll(self.
allocatedRole.applies -> flatten())

```

This constraint is also defined in the **context** *Agent* and asserts that the *Role* allocated to an *Agent* only includes *Capabilities* the particular *Agent* can perform. It must be monitored at runtime to ensure that an agent can perform the task it is supposed to do at any given time. If this constraint does not hold anymore, the system has to be reconfigured. As the constraint only uses data that is locally known to the agent, it can be monitored by the agent itself.

2.4 The Restore-Invariant-Approach

What’s so special about the modeling approach presented above? The answer to this question is, that the designer is now able to very precisely specify when a reconfiguration shall happen and which effects a (successful) reconfiguration shall have. This is basically achieved by the OCL constraints and the intuitive understanding that at all times the current role allocation shall be consistent with the OCL constraints. Whenever this does not hold, the system is expected to reconfigure itself such that the role allocation is again valid. It is thus a good idea to make use of this precise specification and to provide a reconfiguration mechanism which directly takes the specification as input data.

From a more formal point of view an Organic Computing system may be seen as a (possibly infinite) set of traces, where each trace represents one possible run of the system. A single trace is then the sequence of all states the system will assume during one run. The constraints presented above can thus be understood as a partitioning of states into “good” states (where the role allocation is correct) and “bad” states (where the role allocation is incorrect). An Organic Computing

system is then a system which continuously monitors these constraints and *autonomously* tries to change itself (i.e. its role allocation) such that the constraints are valid again. This very generic approach is called *Restore-Invariant-Approach* (RIA). The Restore-Invariant-Approach distinguishes productive states in which the system performs its normal operations and reconfiguration states where a new role allocation is acquired and established. Whenever functional properties are not met (i.e. the system is not productive), a reconfiguration will be started which will reconfigure the system such that these properties are met again (i.e. it can be productive again). This intuitive notion of a RIA is sufficient to understand the remainder of this paper. For a more formal definition of the principles of RIA, please refer to [3] and [4].

An ODP-class system is typically defined through finite sets of agents, capabilities and admissible tasks. As described above, each agent has a number of roles that were allocated to it. Because roles are basically tuples of multiple agents, capabilities and tasks the set of roles and also the set of theoretically possible role allocations are finite. As a consequence it is now possible to use a generic, off-the-shelf SAT solver for finding valid role allocations whenever possible. The following section shows how this could be done using the tool “Kodkod” [14] for this task.

3 Expressing ODP in Relational Logic

This section first introduces an abbreviated version of the Kodkod syntax and an informal semantics. Using the syntax as a basis, the second subsection then describes how a relational model can be derived from an ODP models. Together this allows to use Kodkod as universal reconfiguration algorithm ODP-class systems.

3.1 Relational Logic

The relational logic used in Kodkod is a core subset of the Alloy modeling language [5]. Tables 1 and 2 show extracts of the abstract syntax of the Kodkod logic and the informal semantics.

For relations the basic set operators are available with their standard set theoretic meanings. In addition to the basic logical operators (or, and, negation), both modal quantifiers (all and exists) with their standard interpretation can be used. Further, cardinality operators *some*, *one*, *no*, and *lone* allow expressions

Table 1. Extract of abstract syntax

expr :=		formula :=	
expr + expr	union	expr in expr	subset
expr . expr	join	all varDecl formula	universal
expr × expr	product	some varDecl formula	existential
relVar	relation		
quantVar	quantified variable		

Table 2. Extract of abstract syntax

```

problem := univDecl relDecl* formula
relDecl := relVar :arity [constant, constant]
univDecl := { atom[, atom]* }
varDecl := quantVar : expr
constant := { tuple* }
tuple := (atom[, atom]*)

```

about the amount of tuples within a set. For a complete syntax and a formal semantics please refer to [13].

$$seq := \langle atom[, atom]^* \rangle$$

To allow a more readable notation of the model and the employed formulas we use the following notation for sequences (and also allow the use of sequences within tuples)².

A **Kodkod problem** defines the input for Kodkod. It consists of a *universe* declaration containing a set of *atoms* (identifiers), a set of *relation declarations* and a *formula*. Typically, some relations are under-specified. These relations then appear as variables in the formula. The goal is to find a restriction of relations such that the formula is *satisfied*.

The **core idea** of the relational model is to (1) derive the universe from the object model of the system, (2) derive (fully specified) relational declarations from the current system state (for example associations between objects) and (3) define a (fully under-specified) relation for *allocatedRoles*. The OCL constraints (which basically describe what are correct allocations of roles) are transformed into a relational formula (all constraints are combined by logical conjunction). Together, this allows to reduce the problem of finding a correct role allocation to a constraint solving problem, which can be solved automatically by Kodkod.

3.2 Kodkod Model

In this section the formal representation of the Organic Design Pattern with relational logic presented above is described.

Universe declaration. The first step is to model the ODP concepts. Concepts correlate to sets of unique identifiers, for each instance of a concept. For the task of reconfiguration not all concepts, respectively their instances, are needed. Basically the concepts *Agent*, *Capability*, *Task* and *Role* are required. This results in three sets of atoms – *Ag*, *Cap* and *R*. The union of these sets defines the universe *U* and therefore all atoms which make up the tuples of the relations.

² As Kodkod does not natively support sequences, sequences are resolved by adding atoms to the universe and adding a relation which contains all the elements of the particular sequence. In addition, an order relation is defined to keep the information about the order of the atoms within the sequence.

For a system with n (instances of) agents, m capabilities and p tasks, this will result in the following sets of identifiers:

$$\begin{aligned} Ag &:= \{agent_1, \dots, agent_n\} \\ Cap &:= \{capability_1, \dots, capability_m\} \\ R &:= \{role_1, \dots, role_{n*(m+1)}\} \\ U &:= Ag \cup Cap \cup R \end{aligned}$$

The concept *Task* is formalized as a set of the sequences according to the actual tasks in the system. For example

$$Task := \{(capability_1, \dots, capability_p), \dots\}.$$

As it is build of atoms it is not included in the universe. The number of atoms for roles can be set to the maximum of possible combinations. For efficiency we reduce the number to $n * (m + 1)$ atoms by default. This allows for one role for each agent and each capability and an additional default role, where no capability should be applied and resources should only be passed to another agent.

Relational Declarations. The next step is to define the relational declarations *relDeclarations* needed for the problem. As Kodkod logic is untyped, we define one unary relation for each of the sets *Ag*, *Cap*, and *R*. The relations *Agent*, *Capability*, *Task* contain all elements of their according sets as singletons. This allows to distinguish the atoms later on.

The next step is to formalize the associations of ODP. For each n-ary association we create an n-ary relation bound to the according set of instances. Here we need to distinguish two types of associations. The ones that are describing system state and may not change during reconfiguration and the ones that may be changed. The unchangeable associations are *has*, *input*, *output* as they describe the system state. Those are bound exactly (lower and upper bound are equal) to the tuples derived from the system state.

Table 3. Relations describing the system state

relVar	e.g. [exact bound]
has : ₂	$\{(ag_1, capability_1), (ag_1, capability_2), \dots\}$
input : ₂	$\{(ag_1, ag_2), (ag_1, ag_3)(ag_2, ag_1), \dots\}$
output : ₂	$\{(ag_1, ag_2), (ag_1, ag_1)(ag_2, ag_1), \dots\}$

The changeable relations are the ones that can be evaluated during the re-configuration. These relations are upper bound to the unary relations defined for each concept. The lower bound is the empty set. Table 4 below shows the relations and their bounds. *seq* is an abbreviation for a sequence and denotes a subsequence of *Task*. This is resolved by additional atoms and in the Kodkod model. This reflects the $\{ordered\}$ annotation in ODP.

Table 4. Free relations for reconfiguration

relVar	[lower bound, upper bound]
allocatedRole : ₂	[{}, Agent × Role]
precondition-task : ₂	[{}, Role × Task]
precondition-state : ₂	[{}, Role × seq(Capability)]
precondition-port : ₂	[{}, Role × Agent]
applies : ₂	[{}, Role × seq(Capability)]
postcondition-task : ₂	[{}, Role × Task]
postcondition-state : ₂	[{}, Role × seq(Capability)]
postcondition-port : ₂	[{}, Role × Agent]

The difference between upper and lower bound defines the degree of flexibility the system has and which can be used during reconfiguration to find a satisfying evaluation. Satisfying evaluations are evaluations that fulfill constraints formulated on this relational model which are described in the next section.

Formulas. Section 2.3 describes how system behavior is specified on ODP level using OCL constraints. For reconfiguration, these constraints are formalized in the relational logic of Section 3.1. The OCL *dot* operator ('.') equals the *join* operator '·' in the relational logic. The other used OCL constructs are formalized as follows:

$$\begin{aligned}
C_2 \text{ in } C_1 & \quad : \Leftrightarrow C_1 : \text{Collection}(T) \rightarrow \text{includesAll}(C_2 : \text{Collection}(T)) \\
\{c\} \text{ in } C & \quad : \Leftrightarrow C : \text{Collection}(T) \rightarrow \text{includes}(c : T) \\
(\text{all } c : C \mid \text{formula}) & \quad : \Leftrightarrow C : \text{Collection}(T) \rightarrow \text{forAll}(c : T \mid \text{formula})
\end{aligned}$$

As all OCL constraints are invariants (keyword: **inv**) all constraint are quantified with respect to their context (here: Agent). Using the definitions above the constraints *con1* ... *con4* concerning consistency issues are formalized as follows:

$$\begin{aligned}
(\text{con1}) \text{ all } a : \text{Agent} \mid a.\text{allocatedRole}.\text{precondition-port} \text{ in } a.\text{input} \\
(\text{con3}) \text{ all } a : \text{Agent} \mid \text{all } b : a.\text{allocatedRole}.\text{postcondition-port} \\
\quad \mid \{a\} \text{ in } b.\text{allocatedRole}.\text{precondition-port}
\end{aligned}$$

The monitoring constraint *mon1* is formalized accordingly:

$$(\text{mon1}) \text{ all } a : \text{Agent} \mid a.\text{applies} \text{ in } a.\text{has}$$

As already mentioned, there might be more constraints needed to ensure functional properties. For example, properties that ensure that at least one agent is finishing the task or that the roles of one agent contain only the same capability to apply, as agents should specialize on one capability or switching capabilities is expensive.

The conjunction of all these constraints defines the overall formula (*INV*) that needs to be satisfied. The *problem* *P* that the constraint solver should solve is then defined as follows:

$$P := U \text{ relDeclarations } INV$$

4 Case Study

The following case study illustrates how the proposed reconfiguration algorithm can be employed in the domain of production automation systems. The example describes an autonomous production cell the way it could be designed in the future. Traditional engineering handles production cells in a very static way. Individual machines that process a workpiece are linked with each other in a strict sequential order by conveyors or similar mechanisms. The layout of the cell is therefore pre-defined, very inflexible, and rigid. Additionally – and maybe most importantly – such a system is extremely error-prone as failure of one component will bring the whole system to a standstill.

The vision of an autonomous production cell presented in this paper consists of robots which are capable of using different tools and which are connected with flexible, autonomous transportation units. The functional goal of the cell is to process workpieces according to a given specification, the *task*. A task is a user-defined sequence of tool applications.

The example system considered here contains three robots and two autonomous carts. Each robot has three distinct capabilities: drill a hole in a workpiece, insert a screw into a drilled hole and tighten an inserted screw. In the standard scenario, every workpiece must be processed by all three tools in a given order (1st: drill, 2nd: insert, 3rd: tighten = DIT). Changing the tool of a robot is very time consuming compared to the time required to perform the task itself. Therefore the initial configuration of the system is to distribute the task among the robots, such that no robot has to switch tools and organize workpiece transportation accordingly. This situation is shown in Figure 2. The first robot drills a hole in the workpiece, the second one inserts a screw and the third one tightens this screw.

4.1 The ODP-Based Design

The first step of the modeling process is to instantiate the Organic Design Pattern and map its concepts to the domain. The production cell comprises twotypes

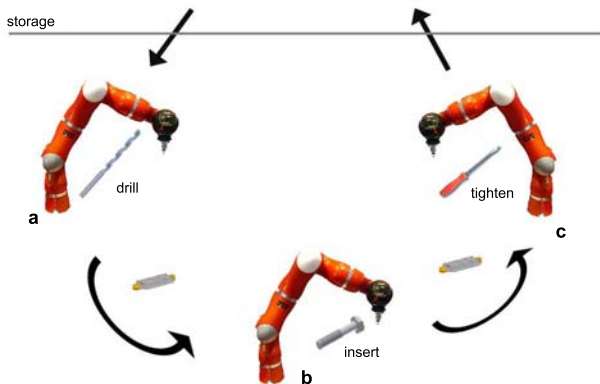


Fig. 2. Adaptive production cell

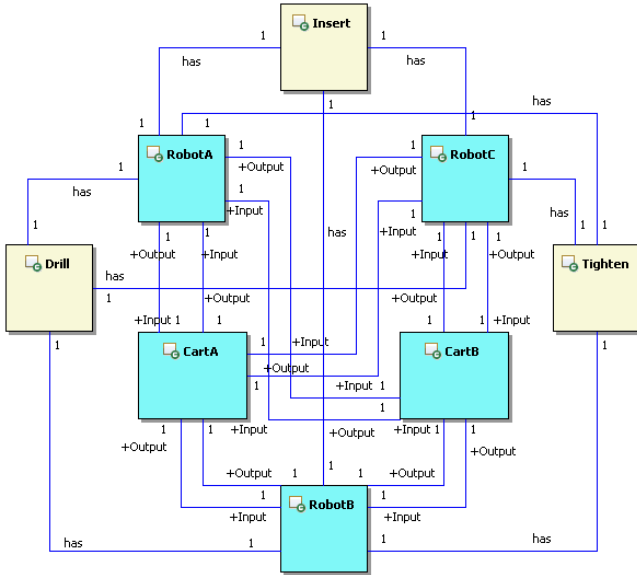


Fig. 3. Excerpt of an instance of an autonomous production cell

of *Agents – Robots* and *Carts*. The capabilities of a robot are *Drill*, *Insert* and *Tighten*, whereas an autonomous cart has no special capability. The fact that every robot has every tool adds redundancy and therefore a degree of freedom to the system. Due to the nature of the system, *Workpieces* (instances of *Resource*) can only be given from *Robots* to *Carts* and vice versa. This is captured by restricting *Input* and *Output* associations (so there is no *Input/Output* association between carts).

Figure 3 shows an excerpt of the instantiation of the ODP with three robots and two carts. Here, the optimal case with full capabilities and functionality is depicted. For better readability, capabilities are represented as data types, while robots and carts are object representations.

During runtime, the object model typically changes, whenever (external) disturbances occur. A broken tool e.g. will result in a change of the *has* association, changes of available agents will result in additional or removed agent objects and new workpieces (with different tasks) will result in additional task objects.

In addition to refinement of the pattern, additional OCL constraints arise out of domain-specific requirements. One example for the domain of production automation is that every agent shall (whenever possible) participate in the process. This property is captured by the following OCL constraint:

```
specific1 inv: self.allocatedRole -> size() > 0
```

This constraint really is domain-specific. For example, in an energy-efficient sensor network it not all agents should participate all the time but rather take turns

in relaying data. This would of course lead to different constraints. However, note that such constraints describe *domain-specific* properties and are established at design time. Thus they are not subject to change during runtime.

4.2 The Relational Representation

The instantiation which describes the static part of the system presented above is now transformed into the relational model as described in Section 3.

Universe declaration: As mentioned in Section 3.2 the first step towards a relational model is the definition of the universe (i.e. Agents, Capabilities and Roles). In the example this will yield:

$$\begin{aligned} Ag &:= \{robot_a, robot_b, robot_c, cart_a, cart_b\}. \\ Cap &:= \{d, i, t\} \\ R &:= \{R_1, \dots, R_{20}\} \end{aligned}$$

Note, that here the suggested limit of $n * (m + 1)$ roles is being used. Analogously the set of all task $Task := \{ \langle d, i, t \rangle \}$ (drill-insert-tighten) is defined.

Relational Declarations: The next step is to define the relational declarations *relDeclarations* for this application. ODP-specifics relations are (obviously) identical to the definitions of Section 3.2. For example for the relation *Agent* this would be $Agent :_1 \{(robot_a), (robot_b), (robot_c), (cart_a), (cart_b)\}$.

The other relations are created analogously. Domain-specific relations are derived from the domain-specific model of the system (see Section 4.1). Some examples are shown in Figure 3:

$$\begin{aligned} has &:= \{ (robot_a, d), (robot_a, i), (robot_a, t), \\ &\quad (robot_b, d), (robot_b, i), (robot_b, t), \\ &\quad (robot_c, d), (robot_c, i), (robot_c, t) \} \\ input &:= \{ (robot_a, cart_a), (robot_a, cart_b), \\ &\quad (robot_b, cart_a), (robot_b, cart_b), \\ &\quad (robot_c, cart_a), (robot_c, cart_b), \\ &\quad (cart_a, robot_a), (cart_a, robot_b), (cart_a, robot_c), \\ &\quad (cart_b, robot_a), (cart_b, robot_b), (cart_b, robot_c) \} \\ output &:= input \end{aligned}$$

Note, that these may also automatically be generated during runtime, as long as the system can reflect upon it's state (i.e. what Agents are present, what Capabilities do they have, etc.).

Formulas: In addition to the generic formulas derived from the OCL constraints, we need to formalize the application specific constraint *specific1*. This is expressed by the following formula:

$$(specific1) \text{ all } a : Agent \mid somea.allocatedRole$$

Adding this formula to the generic formula will lead to the formula the constraint solver should find a solution for.

4.3 Reconfiguration

Reconfiguration is done by finding an evaluation of the remaining relations (e.g. *allocatedRole*) that satisfies the constraints. For each reconfiguration the relational model is adapted to the current system state and the constraint solver is started with this changed model to find a solution. A possible evaluation for the initial configuration of the relation *allocatedRole* and its following relations is shown in Table 5.

If, for example, a failure like a broken tool occurs (e.g. a broken drill at *robot_a*), the corresponding relation *has* will change to *has*/ $\{(robot_a, d)\}$ which means that a change in *relDeclarations* occurred. As a consequence, constraint *mon1* stating that only available capabilities are assigned within roles will evaluate to *false* and a reconfiguration is triggered again. The will then enter a reconfiguration phase use Kodkod to calculate a new evaluation for *allocatedRole* with respect to the other values of the relations and go back to production again.

A special case has to be considered if an agent fails entirely. Assume, e.g. that *robot_c* is no longer available and only two robots are left to perform the task. This corresponds to a change in the set *Ag* and thus a change in the *universe*. In such a case, systems designed with ODP support *graceful degradation* where as much functionality as possible is retained, even if processing times deteriorate. In the example, one robot will be assigned several roles and thus performs two actions on each workpiece. This way the workpieces are still processed correctly but as switching tools requires a lot of time compared to applying the tool, the workpieces will remain in the cell longer and throughput decreases. The role allocation for the described situation is depicted in Table 6.

Table 5. Initial role allocation

allocatedRole		precondition				postcondition		
Agent	Role	port	state	task	applies	port	state	task
<i>robot_a</i>	<i>role₁</i>	\emptyset	$\langle \rangle$	$\langle d, i, t \rangle$	$\langle d \rangle$	<i>cart_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$
<i>cart_a</i>	<i>role₂</i>	<i>robot_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$	$\langle \rangle$	<i>robot_b</i>	$\langle d \rangle$	$\langle d, i, t \rangle$
<i>robot_b</i>	<i>role₃</i>	<i>cart_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$	$\langle i \rangle$	<i>cart_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$
<i>cart_b</i>	<i>role₄</i>	<i>robot_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$	$\langle \rangle$	<i>robot_c</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$
<i>robot_c</i>	<i>role₅</i>	<i>cart_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$	$\langle t \rangle$	\emptyset	$\langle d, i, t \rangle$	$\langle d, i, t \rangle$

Table 6. Role allocation for graceful degradation

allocatedRole		precondition				postcondition		
Agent	Role	port	state	task	applies	port	state	task
<i>robot_a</i>	<i>role₁</i>	\emptyset	$\langle \rangle$	$\langle d, i, t \rangle$	$\langle d \rangle$	<i>cart_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$
<i>cart_a</i>	<i>role₃</i>	<i>robot_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$	$\langle \rangle$	<i>robot_b</i>	$\langle d \rangle$	$\langle d, i, t \rangle$
<i>robot_b</i>	<i>role₄</i>	<i>cart_a</i>	$\langle d \rangle$	$\langle d, i, t \rangle$	$\langle i \rangle$	<i>cart_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$
<i>cart_b</i>	<i>role₅</i>	<i>robot_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$	$\langle \rangle$	<i>robot_a</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$
<i>robot_a</i>	<i>role₂</i>	<i>cart_b</i>	$\langle d, i \rangle$	$\langle d, i, t \rangle$	$\langle t \rangle$	\emptyset	$\langle d, i, t \rangle$	$\langle d, i, t \rangle$

In another scenario, the task is changed during runtime. If, e.g., three holes need to be drilled, the system self-adapts to the new task. This is expressed in a change of the task relation: $task = \{([d, d, d])\}$. A reconfiguration cycle is automatically started by the system to configure the robots in a way that the new task is performed.

4.4 Implementation

The presented reconfiguration algorithm is employed in association with an implementation of the case study with a generic framework for systems modeled with ODP which is based on the multi-agent system Jadex [1]. The robots and carts are implemented as agents and perform the “drill, insert, tighten” sequence as described above. When one of the robots loses one of the capabilities it needs to perform one of its roles, a reconfiguration is started. The VirtualCentralizedO/C is informed about the failure, stops the processing of resources and gathers the individual agent’s configuration. This global view of the system is then transmitted to the reconfiguration algorithm via a generic web service interface. The Kodkod implementation is coupled with a web service endpoint which accepts the data and transforms it into the relational model required. Kodkod calculates the new role allocation for this model and the result is transformed again and returned to the ODP system as a reply to the original web service call. The O/C distributes the new role allocation to the agents and starts the system again. Our experiments showed that the transformations and the calculation of roles work very well. For small to medium sized applications runtime is not an issue (a complete reconfiguration is done within seconds) and the coupling with the multi-agent system is reliable and simple to use. In the next step, we plan to couple the entire system with a graphical simulation of the production cell.

5 Conclusion

Design and construction of Organic Computing systems is a challenging task. There already exist some approaches to help during design by suggesting a specific architecture [2, 7, 12]. Reconfiguration during runtime, however, is either left out or only specified on a very high level. There are approaches to configure systems with constraint satisfaction methods like the one in [15] but these are limited to constraints about system structure and not about system dynamics.

This paper proposes a universal reconfiguration algorithm for role-based Organic Computing systems. The approach has been refined to a running implementation for the class of ODP systems. The core idea is to use design artifacts (which capture the specification of reconfiguration) and transform them into a constraint solving problem. The presented implementation uses the Kodkod constraint solver. Future work will include investigations on how to incorporate system constraints into the design process more closely as in, e.g., the Darwin ADL [8]. Another open issue is the question how to deal with quantitative requirements. These often play a key role for self-optimizing systems and it is a

challenging task to translate them into a constraint solving problem. We plan to investigate how cost functions can be integrated into our approach.


References

- [1] Braubach, L., Pokahr, A., Lamersdorf, W.: *Jadex: A BDI Agent System Combining Middleware and Reasoning*. Software Agent-Based Applications, Platforms and Development Kits (2005)
- [2] Georgiadis, I., Magee, J., Kramer, J.: Self-organising software architectures for distributed systems. In: *Proceedings of the first workshop on Self-healing Systems*, pp. 33–38. ACM Press, New York (2002)
- [3] Güdemann, M., Ortmeier, F., Reif, W.: Formal modeling and verification of systems with self-x properties. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) *ATC 2006*. LNCS, vol. 4158, pp. 38–47. Springer, Heidelberg (2006)
- [4] Güdemann, M., Nafz, F., Ortmeier, F., Seebach, H., Reif, W.: A specification and construction paradigm for organic computing systems. In: Brueckner, S., Robertson, P., Bellur, U. (eds.) *Proceedings of the Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, pp. 233–242. IEEE Computer Society Press, Los Alamitos (2008)
- [5] Jackson, D.: Alloy: a lightweight object modelling notation. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 11(2), 256–290 (2002)
- [6] Kephart, J., Chess, D.: The vision of autonomic computing. *IEEE Computer* (January 2003)
- [7] Kramer, J., Magee, J.: Self-Managed Systems: an Architectural Challenge. In: *International Conference on Software Engineering*, pp. 259–268. IEEE Computer Society, Washington (2007)
- [8] Magee, J., Dulay, N., Eisenbach, S., Kramer, J.: Specifying Distributed Software Architectures. In: Schäfer, W., Botella, P. (eds.) *ESEC 1995*. LNCS, vol. 989, pp. 137–153. Springer, Heidelberg (1995)
- [9] Müller-Schloer, C.: Organic computing: on the feasibility of controlled emergence. In: *CODES+ISSS 2004: Proceedings of the 2nd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pp. 2–5. ACM, New York (2004)
- [10] OMG. *Object Constraint Language, OMG Available Specification* (2006)
- [11] Richter, U., Mnif, M., Branke, J., Müller-Schloer, C., Schmeck, H.: Towards a generic observer/controller architecture for Organic Computing. *Informatik*, 112–119 (2006)
- [12] Seebach, H., Ortmeier, F., Reif, W.: Design and Construction of Organic Computing Systems. In: *Proceedings of the IEEE Congress on Evolutionary Computation 2007*. IEEE Computer Society Press, Los Alamitos (2007)
- [13] Torlak, E., Jackson, D.: *The Design of a Relational Engine*. Technical report, Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory (2006)
- [14] Torlak, E., Jackson, D.: Kodkod: A Relational Model Finder. In: Grumberg, O., Huth, M. (eds.) *TACAS 2007*. LNCS, vol. 4424, pp. 632–647. Springer, Heidelberg (2007)
- [15] Warren, I., Sun, J., Krishnamohan, S., Weerasinghe, T.: An Automated Formal Approach to Managing Dynamic Reconfiguration. In: *21st IEEE/ACM International Conference on Automated Software Engineering, ASE 2006*, pp. 37–46 (2006)

Towards Self-organization in Automotive Embedded Systems

Gereon Weiss, Marc Zeller, Dirk Eilers, and Rudi Knorr

Fraunhofer Institute for Communication Systems (ESK),
Hansastr. 32, 80686 Munich, Germany
{gereon.weiss,marc.zeller,dirk.eilers,rudi.knorr}@esk.fraunhofer.de
<http://www.esk.fraunhofer.de>

Abstract. The rising complexity of upcoming embedded computing systems cannot be managed by traditional methodologies. Hence, with Autonomic Computing and Organic Computing new paradigms of self-organizing systems have been introduced. The automotive sector as an application domain of embedded systems also has to face the challenge of growing complexity. Thus, in this paper we show a potential evolution of automotive electronic systems to (partly) self-organizing systems and identify the nowadays missing capabilities. We chose the infotainment domain for exemplarily demonstrating this evolution by introducing new enhancements for self-organization. These extensions are evaluated in a case study of a typical vehicle infotainment system based on the MOST (Media Oriented Systems Transport) technology. The implementation and evaluation show that the newly introduced techniques work well in a realistic scenario. Thereby, we validate that an evolution of present statically designed automotive electronic systems to self-organized systems is feasible 

1 Introduction

Nowadays certain software-driven embedded systems have reached an enormous degree of complexity and penetrate everyday life. The growing percentage of functionalities lead to more and more unmanageable systems. Additionally, the distribution of such systems cause an even greater challenge to address. An example for the growing complexity is the automotive sector with its big share of the embedded systems market. Car manufacturers (Original Equipment Manufacturers - OEM) integrate a large distributed system developed separately by numerous suppliers. With such rising complexity the need for dependability and safety still remains. Thus, automobiles constitute distributed embedded systems with large heterogeneity and very varying requirements. Safety related systems with hard real-time constraints as well as entertainment functionalities with soft real-time or without any real-time constraints coexist. Currently this

¹ This work has been supported by the project DynaSoft funded by the bavarian Ministry of Economic Affairs, Infrastructure, Transport and Technology.

fact is addressed by a divide-and-conquer strategy: a separation into domains where each subsystem is specifically suited for its application. But with the actual increasing interdependency of these systems the strict separation cannot be maintained. Moreover, the variability and customization for individual cars are growing. Software-based innovations are foreseen to be the predominant factors in new automobiles over the next years. The inherently growing complexity of designing and managing dependable and secure future automotive electronic systems becomes a critical issue [1]. Furthermore, the long-term life-cycle of a vehicle enables the up-and-coming field of after-market products which comprises of the integration of consumer devices, new devices or software services, to name just a few. Those trends pose a great challenge to the OEM - historically being car builders not software engineers. To bear the challenge of upcoming complexity within the IT-architecture new paradigms have to be explored.

Within the last years the field of self-organization has proven to be a promising solution for the management of systems which are unmanageable with traditional methods. The system is managed within control loops [2]. This can be realized by single control loops or even distributed control loops at different layers creating the phenomenon of *emergent behavior* [3]. The systems state is monitored continuously within such a control loop. Deviations of the measured properties are analyzed and control actions to correct the systems state are planned. Systems being managed in such a way can self-organize and possess the ability to adapt to their environment.

The features of *self-organized systems* might also provide a solution for the growing complexity in the automotive domain. Within this work we examine the feasibility and opportunities of integrating self-organizing techniques into vehicles. For this purpose we present a potential evolution of automotive electronic systems to self-organized systems. Thereby, challenges of these systems towards self-organization are outlined. Afterwards, we focus on the automotive infotainment area and introduce new enhancements in this domain for self-organization. We implement and evaluate the introduced extensions in a case study of a typical today's infotainment network.

The structure of this paper is organized as follows: Firstly, the related work in reference to self-organizing and automotive systems plus our novel contributions are described in Sect. 2. Afterwards a potential evolution of automotive electronic systems towards self-organized systems and challenges in today's in-vehicle systems are presented in Sect. 3. In Sect. 4 we outline as field of application the automotive infotainment technology and our enhancements for self-organization. A case study of the newly developed infotainment self-organization enhancements, its implementation and evaluation are presented in Sect. 5. Finally, we conclude the paper in Sect. 6.

2 Related Work

In current research self-organizing systems have been identified and addressed as promising solution for future adaptive and self-managing information technology.

Multi-faceted research is being pursued in this research field [4]. Various systems are enabled to provide so-called *self-x properties* [5] like Self-Protection, Self-Healing, Self-Configuration among others. The pursued goal is to reduce the management or control from the outside of elements by applying inner self-control mechanisms. Also the interaction and the achievement of an intended behavior of such self-managing elements is of concern. In the following we will present a non-exhaustive list of approaches utilizing self-organization in general and in the automotive domain.

One of the first approaches is driven by IBM - the Autonomic Computing Paradigm [6]. The main idea is the adaptation of the behavior of the Central Nervous System which interacts autonomously. As basic principle the management of *autonomic elements* is represented by a reconfiguration-cycle where each autonomic element monitors and analyzes the environment, plans its next steps and executes the planned actions. Originally, the focus lies on the management of large computer networks. With *Organic Computing* [7] a novel principle for self-organizing systems is given by imitating adaptive, life-like behavior in the nature. Self-organization is realized on different abstraction levels with observer/controller models utilizing control-loops. No particular field of application is addressed and interdisciplinary research is covered. With the Self-adaptive Software Program [8] a very ambitious research field is addressed where software evaluates and changes its own behavior at runtime. Therefore descriptions of intentions and alternative behavior need to be added in shipped software.

2.1 Self-x in the Automotive Domain

In the automotive sector several initiatives have already focused on evaluating self-organizing techniques for vehicles. A high-demanding goal for the future of transportation are autonomous cars which can adapt even in high complex scenarios as in urban traffic [9]. As promising as early results are, many - not only technical - problems are not solved yet and thus the future of autonomous driving is still not foreseeable yet.

For the in-vehicle information and entertainment functionality the MOST bus is a widespread established standard. It facilitates functional composition with a powerful API and already features with its configuration management very limited self-x properties. A more detailed introduction is outlined in Sect. 4.1. The Automotive Open System Architecture (AUTOSAR) [10] initiative is a consortium with the goal of an open standard for automotive software architecture. Through a component-based architecture the reuse and scalability of future automotive software is pursued. By a virtual integration of software components (Virtual Function Bus) the allocation of functionality to ECUs (Electronic Control Units) can be assembled at design time. Even though this approach facilitates a more liberal way of allocation it does not support any dynamic allocation at runtime. Hence, self-organization techniques that rely on reallocation of functions cannot be applied. In [11] self-healing and self-configuration is evaluated in a component-based automotive architecture which indicates the potentials arising with these techniques. Dinkel [12] focuses on the development and simulation of a

completely new IT-architecture for future cars. It utilizes Java and OSGi for simulation purpose and is not applied in the field. The ongoing DySCAS project [13] focuses on developing a middleware enabling dynamic self-configuration in today's cars. The upcoming results will have to show if the benefits justify the drawbacks of the overhead for a middleware. Within the research project ReCoNets [14] fault-tolerance is addressed by bringing Hardware/Software-Reconfiguration into the automobile. Although reallocation of both hardware and software is a consequent progression of the currently advancing adaptivity and decomposability, it is not aligned with present automotive development method (e.g. FPGA reconfiguration) but might be feasible in the future. As briefly described here different approaches are in progress enabling self-x properties in future cars with various degrees of possible adaptation. Many open challenges need to be researched for meeting the domain-specific requirements of automotive electronic systems (e.g. the verification of adaptation). Though no project focuses on the embedding of techniques in present automotive electronic systems allowing a transition to self-organizing systems.

2.2 Our Contributions

The need for a change of actual development paradigms to more *adaptive systems* is becoming obvious by the arising complexity challenges of modern embedded systems. Because future systems will not likely be developed again from zero but will base on present and prior standards and systems, modern systems have to evolve to self-organizing systems. To the best of our knowledge we outline a novel anticipated *evolution of automotive systems* to self-organizing systems with higher degree of inner variability. Additionally, we identify challenges in the development of today's automotive electronic systems for allowing the application of self-x technologies. We present a domain-based approach which envisages a per domain multi-level/hybrid architecture that considers the varying requirements of the different domains and services. As an example for the extension of a modern system to provide self-x features we propose new additions to a modern automotive infotainment network. In a case study of a today's automotive multimedia system we validate the proposed extensions. Thus, we show that the evolution of automotive electronic systems towards self-organized systems can be realized.

3 Evolving Self-organizing Automotive Electronic Systems

Today's development process of automotive electronic systems is characterized by the original car development. Thus, systems are *statically* designed from scratch to fulfill the requirements. Based on a requirement analysis a specification is derived. Thereafter the partitioning of functions in hard- and software is performed in an early stage. The functionalities are distributed in different *in-vehicle domains* (e.g. body, power train or infotainment). Even though this approach

facilitates the determination of meeting safety and security requirements, it constrains a system's possibilities to adapt at runtime. Efforts to embed more dynamics to the development process (like product line variability) are nowadays restricted to the design phase. As prevalent example the possibility to compose functions at design time - which is offered by the above mentioned AUTOSAR specification - allows for a higher degree of variability in the designers' choice. This enables a greater potential for static system optimizations. But inherently with these upcoming degrees of freedom the complexity of managing these systems grows. New control mechanisms have to be found to keep security aspects and error rates low. A trend can be identified in current in-vehicle electronic systems to incorporate internally controlled variability at different levels of abstraction (e.g. intelligent sensors or the MOST network itself). A central, decentral or multi-level observer/controller model envisaged by the Organic Computing paradigm where the user as final controlling instance can also be developed for the management of vehicles. To derive the *evolution* of automotive systems to self-organized systems the definitions of self-organization presented in [15] are considered. With the static definition we can deduce a roadmap for the evolution of self-organization in automotive systems. The complexity reduction R is thereby defined as:

$$R = V_i - V_e \quad (1)$$

where V_i is the internal and V_e the external variability of the object of investigation. From this the static degree of complexity S is developed as:

$$S = \frac{R}{V_i} = \frac{V_i - V_e}{V_i}, \quad (0 \leq S \leq 1) \quad (2)$$

Hence, we devise an evolution of self-organization in the automotive sector and picture it in comparison to the inner variability of automobile (sub-)systems or components. As the variability of these systems is expected to grow with the progression of time because of the increasing interacting functionality, the evolution of automotive self-organization is depicted.

As we can derive from Fig. 1 today's vehicles in the field are equipped with highly specific electronic and software components. This keeps the possibility for the reuse of components low and allows only incorporating very low inner variability. Even software product line and component-based approaches cannot ease the need for more degrees of variability, when the underlying paradigm still remains encapsulating functionalities by one supplier in one ECU, establishing a dedicated specific service. Therefore we try to foresee the following recognizable phases as shown in Fig. 1 in the automotive development:

- Proprietary software and hardware was vendor-specific and ready-made systems were shipped. Software was implemented architecture-dependent.
- AUTOSAR enables a more flexible reuse of software components and paves the way for more design time variability. The possible composability of this approach will lead to more diversity of system implementations and new software-depending services over the next years.

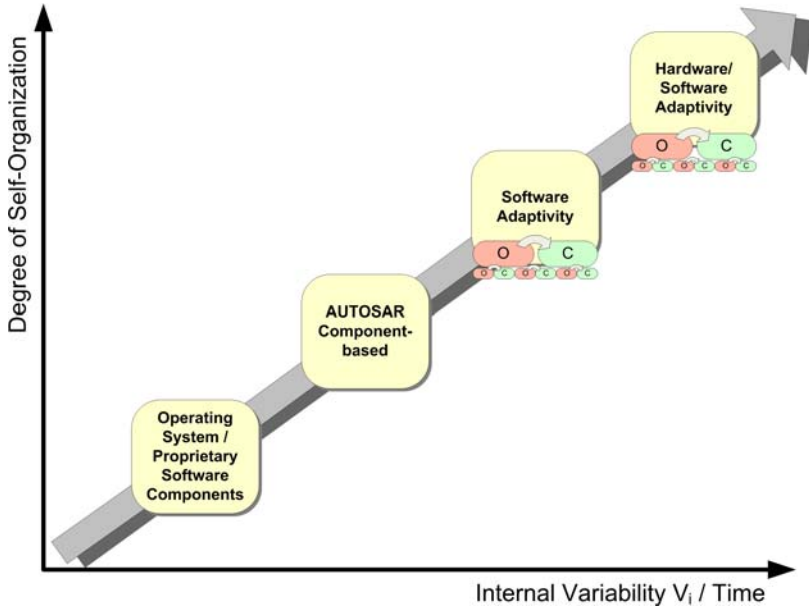


Fig. 1. Evolution of Self-Organization in Automotive Systems

- Because of the arising complexity and the need to fill the management gap of the upcoming complex embedded systems, more and more dynamic self-adaptive software will be integrated in automobiles. Techniques, like for example the reallocation of software, will soften the partitioning of the system into domains and enable a new potential of optimization and robustness.
- With software as promising enabler for runtime car adaptation the potential of hardware evolving to *reconfigurable systems* is realized. By this the boundary of hardware and software vanishes in the design and at runtime. Hence, a dynamic adaptability of hardware and software can be tackled.

Since we have described potential evolutionary steps towards self-organizing vehicles we outline the challenges by an analysis of today's automotive embedded systems. Examining the automotive electronics design we can identify a number of challenges which need to be addressed in order to achieve more self-adaptivity of in-vehicle systems. Generally spoken, the more *design choices* are fixed, the smaller is the achievable degree of freedom for any adaptation to the environment. Consequently, for the evolution to self-organizing automotive electronic systems the following crucial influencing parameters have been identified:

- **Seamless fine-granular functional decomposition:** In modern systems a single ECU is dedicated for a specific functionality. However, with a more modular approach like AUTOSAR there is the need to decompose services into *atomic functions*. This enables the reuse of functionality as it reduces the overhead by erasing the redundancy. Thereby, more freedom for runtime

adaptation arises beneath the great optimization potential if not every service is implementing the same functionality by itself.

- **Abstraction from input, processing and output:** Although sensors and actuators are separated from the computation there is still the necessity for locality of the functions to access the sensor/actuator data. Allowing a more flexible distribution of functions is mandatory to tap the full optimization potential of self-organization. Techniques like publish/subscribe and distributed data access might ease this problem.
- **Abstraction from heterogeneity:** Diverse technologies are incorporated in heterogeneous distributed embedded systems like automotive electronic systems. Only an abstraction from the underlying technology (e.g. via a runtime environment or middleware) will allow the interaction of the components and thus the self-organization of the overall system.
- **Dependability and safety:** In the automotive domain several applications with divergent safety requirements (specified as *Safety Integrity Level*, SIL) are composed to one system. Presently, the requirements are met by separation into domains. Thus, a major challenge is to guarantee and meet the safety requirements of automotive systems even in adaptive systems (for example the ability to meet hard timing constraints).
- **Runtime resource management:** Within statically designed systems most of the available resources are assigned fix. As runtime adaptation is needed to control the growing complexity a runtime resource and conflict management is inevitable for the controlled configuration of the system (e.g. instead of a statically resolved virtual function bus with fixed port assignments in AUTOSAR, a real function bus with a dynamic scheduling).
- **Self-awareness:** Present automotive systems have no capabilities to describe their properties and requirements so that an observer/controller could not obtain enough information about the systems state apart deduced information. Accordingly, a description of the components has to be made available at runtime. For component-based approaches a self-description generated out of the design seems promising. But a tradeoff between the expressiveness with more potential for self-organization and the overhead of a higher complexity for algorithms has to be done.

In order to explore an automotive system in respect to its ability to evolve to a self-organizing system which considers above challenges, in the following we focus on an in-vehicle infotainment system. Due to the fact that automotive subsystems are separated in domains we envisage a self-organization in each *domain* composed in a greater control loop as multi-level/hybrid approach. Hence, an observer/controller is presumed in each domain. A supervising observer/controller organizes the inter-domain control loop. However, in the next sections our focus lies on the evolution to self-organization in the infotainment domain.

4 Enabling Automotive Infotainment Self-organization

In the previous section we have outlined the opportunities and challenges of the evolution to self-organizing automotive systems. In the following we emphasize

the feasibility and potential of such evolving systems by introducing new enhancements of the present-day vehicle infotainment bus MOST. Particularly, the previously described challenges of self-awareness and runtime resource management are addressed. The *infotainment domain* was chosen because of its lack for the strict safety requirements. Firstly, we describe the basic specification of a MOST bus as relevant for the upcoming case study and evaluation. Afterwards, our additions to the MOST network for a runtime reallocation of functionalities are presented.

4.1 Infotainment Bus MOST

MOST (Media Oriented Systems Transport) is a present standard in the automotive infotainment domain standardized by the MOST Cooperation. Overall audio, video, speech or data are transported. Up to 64 ECUs can physically be connected in a ring topology. The MOST standard protocol is specified through all seven layers of the ISO/OSI reference model. Thus, it provides a powerful API for application developers. Different types of communication are possible: control, synchronous, asynchronous and isochronous data transportation. Generally, the communication is separated in logical channels for control data, streaming data and packet data.

Applications are encapsulated in so-called Function Blocks (*FBlocks*). The same FBlocks can be distinguished by a network-wide unique number called InstanceID (*InstID*). Generally, a FBlock incorporates functions and properties. A publish/subscribe mechanism (so-called *notifications*) allows controllers for observing selected properties of FBlocks. Such controllers are also denoted as Shadows which enable the control of a FBlock and the view on its properties.

A FBlock has a distinct functionality like for example the AudioDiskPlayer FBlock which represents a CD Player. At least one type of specific FBlocks with certain network management functionality (denoted as *FBlock Masters*) are required in every network. A TimingMaster is responsible for the synchronization of the network. The NetworkMaster handles the self-configuration like the logical addressing or the service-discovery. At system start-up or in the case of a network change event the NetworkMaster scans the network by requesting the available FBlocks at every node. Reported FBlocks are stored with their InstID and their logical address in a central database, the *Central-Registry*. Applications can seek for FBlocks by requesting this CentralRegistry at the NetworkMaster. Every ECU in the network has a FBlock called NetBlock which provides functionality that affects the whole device. For example it offers functions for information about the available FBlocks or for the address management of the ECU. For the power management of the MOST ring the PowerMaster is introduced. For audio streaming the ConnectionMaster handling the connection management for the audio streaming and the AudioMaster for the mixing of different audio signals are essential. Additional information about the MOST technology is explained in the next sections where necessary.

4.2 Enabling Self-organization in Automotive Infotainment Systems

The infotainment domain has no strict safety and security requirements. Hence, it is well suited for initially introducing support for self-organization. Because MOST lacks the ability of a *runtime functional allocation* we provide enhancements supporting the reallocation of functions. Hence, below a dynamic reallocation of ECU-resources and relocation of FBlocks is proposed for allowing a higher degree of variability within the infotainment network. Because no change in the distribution of functions at runtime is considered in the present standard, additional mechanisms have to be introduced. To keep the overhead low a straightforward methodology is adopted. New functionality is developed which points out how *minimal additions* can be applied to present systems, achieving a higher degree of variability and thus paving the way for self-organization in future automotive electronic systems.

Present functional decomposition of the infotainment system is still performed device-centric. MOST FBlocks are physically bound to actuators or sensors. To enable the full potential of a variable allocation of resources at runtime the interfaces have to be decoupled from the physical components. Thus, nowadays only functions not physically bound to local resources, such as data-driven functionalities like an address book or a music-database, can be reallocated in the network. For example, the decoding, encoding or mixing of data can be processed independently of the data's origin location.

The following concerns are identified to be crucial for the envisaged software runtime reallocation in the infotainment network. At first, a *description* of the requirements and capabilities of the hardware has to be provided at runtime. Furthermore, functionality for the *reallocation* of software at the ECUs has to be developed. This reallocation is realized by an observer/controller which conducts and decides on a software-reallocation depending on the information of the descriptions of the involved components. A functional overview of the envisaged extensions are shown in Tab. 1. The details of the functions and parameters are addressed later for the convenience of the reader with the case study and its evaluation in Sect. 5 and 5.1.

MOST provides descriptions of static interfaces for each ECU at runtime by its indexing of functions (*FktIds*). To provide runtime information about the system

Table 1. Minimal Extensions for the Reallocation in a MOST Network with the Relevant Parameters

FBlock	Function/Property	Parameter
<i>ReconfigurationBlock</i>		
- Property	ECUDescription	DescriptionInfo, ECUDescription
- Property	FBlocksDescription	(FBlockId, InstId, DescriptionInfo)*, Description
- Function	Reallocation	FBlockId, InstID, Type, Context
- Function	Reconfiguration	FBlockId, InstID, TargetStatus
<i>ReconfigurationMaster</i>		

the approach of a *self-description* of each component (FBlocks and ECUs) utilizing the MOST notification service is chosen. Generally, such a self-description has itself a certain degree of variability and can range from a static to a semantic description. A *dynamic description* is used in the implementation to compromise expressiveness and overhead. Because for decomposability and modularity reasons the descriptions are embedded into the components achieving a self-description. Typically the self-description of an ECU and its components are a matter of the NetBlock as the FBlock for functionality concerning the whole ECU. Anyhow, we introduce the new FBlock *ReconfigurationBlock* allowing an easy standard compliant integration without the need for changing the NetBlock. An instance of the ReconfigurationBlock is located at each ECU which is capable of a reconfiguration.

The reallocation management is implemented by one central observer/controller in the infotainment domain. For this purpose consistent to the present MOST mechanisms a new FBlock called *ReconfigurationMaster* is developed. The ReconfigurationMaster subscribes itself to the self-descriptions of the FBlocks and ECUs. Depending on this information it schedules and conducts the reallocation of FBlocks within the network. Since modelling of the algorithm (cp. [16]) for the observer/controller is not our main focus within this paper we stick to a straightforward reallocation management. This permits to keep the overhead low in a first step towards self-organization in the infotainment domain. For the reallocation of FBlocks we need additional functionalities beneath the management functionality. More precisely each ECU involved in the runtime allocation needs functions to carry out the reallocation. As realization for those a transaction-based methodology considering the so-called *ACID* properties (*Atomicity*, *Consistency*, *Isolation* and *Durability*) is envisaged. Due to the fact that a reallocation concerns the reconfiguration capabilities of an ECU, the additional functionality is provided by the ReconfigurationBlock. By calling the ReconfigurationBlock function *Reconfiguration* the ReconfigurationMaster can start the reconfiguration of specific FBlocks. The ReconfigurationBlock utilizes the *Reallocation* function to migrate FBlocks from another ECU to its own.

A controlled deactivation and activation of a FBlock can be carried out by forcing its inner states. Generally, such a component can be in the states active, passive or quiescent [17]. In the addressed infotainment environment - due to no hard deadlines and self-configuration of the addressing by the NetworkMaster - the system can recalibrate itself to normal operating state. Although a physical relocation of software within such a transaction would be feasible, we only outline the parameters of software-migration and restrict ourself to a reinstantiation of the FBlock at the destination ECU. In case of a migration the runtime-environment, the context and the data of the migrated software should be taken into consideration.

5 Case Study

For validation and evaluation purposes a case study of a MOST scenario exploiting the advantage of self-organization in automotive embedded systems is presented. The implementation and evaluation of this MOST system is described in the next

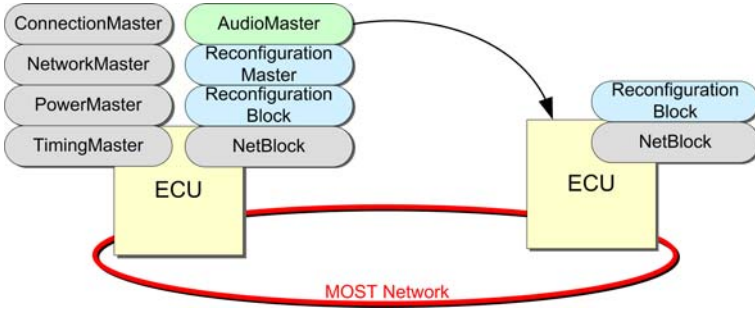


Fig. 2. MOST Network Scenario of the Case Study

section. The following described system comprises of several FBlocks forming a small but entire infotainment unit. In this configuration two ECUs - without loss of generality - are modelled for validating our approach. Besides the standardized FBlocks a ReconfigurationMaster and one ReconfigurationBlock for each ECU are integrated, allowing the *self-organized reconfiguration* of the network. For audio playback the ConnectionMaster and AudioMaster are modelled in the system. The initial distribution of the described system is illustrated in Fig. 2.

At system startup after the standard initialization of the MOST network, the ReconfigurationMaster subscribes itself to the self-descriptions of the ECUs and FBlocks by sending *notification messages* to each ReconfigurationBlock. If an ECU is not capable of reconfiguration and does not possess a ReconfigurationBlock it is not considered as part of the reconfiguration space. This enables and eases the evolution or migration to self-organized systems. The opportunities of such a setup with runtime adaptation can be highlighted by just a few examples. Even the re-allocation of the FBlock is restricted by the need of local resources or the coarse functional decomposition of MOST a self-optimization can optimize the network in respect to certain criterias (e.g. cpu-load, memory consumption, energy consumption, etc.). Also the dynamic generation of a FBlock or the connection of a Consumer Device to the in-vehicle infotainment system benefits from the potential of the reconfiguration ability. If the execution of a FBlock or even a whole ECU fails, the reallocation to another ECU (if the network is still functioning) might maintain the expected functionality of the system.

The latter scenario is evaluated in the next section. An event occurs that the *status* of the AudioMaster changed (because of a malfunction, resource stringency or self-optimization process). Hence, the MOST ReconfigurationMaster as local observer/controller allocates the AudioMaster to another ECU. For this process it has to be relocated within the network. The reconfiguration is carried out by the ReconfigurationMaster utilizing the ReconfigurationBlocks of the involved ECUs.

5.1 Implementation and Evaluation

For the validation of the introduced new techniques for self-organizing automotive systems we have implemented and evaluated the above described MOST case

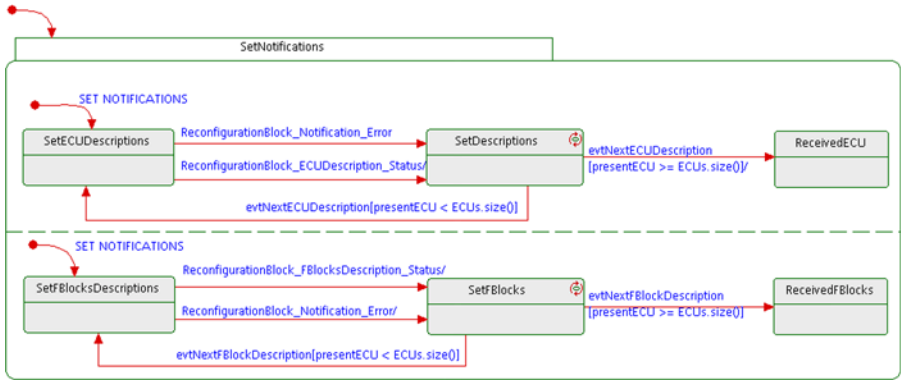


Fig. 3. Initialization Statechart with Notifications

study. A *model-based approach* is utilized because of the already existing frameworks for the automotive domain and the numerous advantages of model-driven development. We use the UML-based (Unified Modeling Language) framework Telelogic Rhapsody which allows the event-based modelling and execution of the systems functionality. The behavior of the FBlocks and Controllers is defined by UML statecharts. The initial network system setup is configured by an UML object model diagram. An additional framework allows the connection to the MOST interface.

Each FBlock and Shadow has been modelled with attributes for a *self-description* (e.g. vendor id, worst case execution time, worst case heap usage). The description content is taken partly from the specification of software attributes from AUTOSAR and completed by missing properties for runtime adaptation, like the ability to be reallocated. Self-descriptions for ECUs (e.g. available DRAM segment size) have also been composed by AUTOSAR attributes for ECUs supplemented by missed attributes for runtime capabilities.

Beneath the standard FBlocks, models for our new introduced ReconfigurationMaster and the ReconfigurationBlock were developed. For demonstration purposes a detail of the Statechart for the ReconfigurationMaster is depicted in Fig. 3 in which the notifications for the self-descriptions of the ECUs and FBlocks are modelled. The ReconfigurationMaster subscribes itself for the self-descriptions of the ECUs and FBlocks. Changes of these self-descriptions can trigger the ReconfigurationMaster to reconfigure the system. The behavior and detail of the control is not subject of this paper but the basis allowing such self-management in a modern automotive network.

In Fig. 4 the startup-sequence of the new developed ReconfigurationMaster is depicted. As shown, the subscription for the different self-descriptions are carried out in parallel. After the registration of the descriptions the ReconfigurationMaster is informed about changes. Local changes recognized at a single ECU result in an *event* triggering the respective ReconfigurationBlock to update the self-description information.

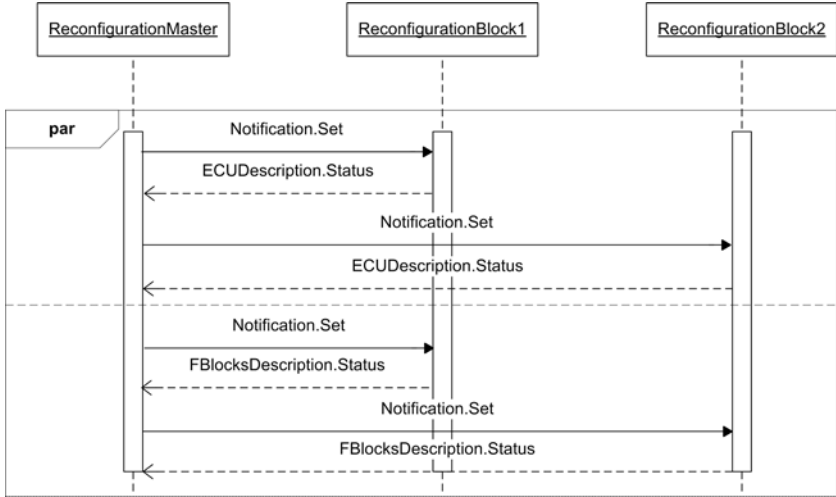


Fig. 4. Initialization Sequence of the ReconfigurationMaster

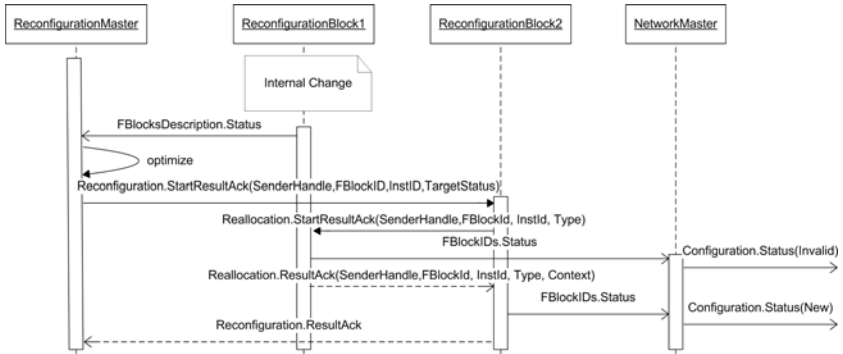


Fig. 5. Sequence Diagram of a Reconfiguration

For the evaluation and validation of the developed new functionalities a failure of a single FBlock (AudioMaster) was simulated. The respective ECU recognized that the AudioMaster cannot be executed anymore (e.g. because the memory is insufficient). This led the ReconfigurationBlock to update the self-description for the AudioMaster. Receiving the new self-description the ReconfigurationMaster reallocates the AudioMaster to another ECU. The sequence of a reconfiguration is illustrated in Fig. 5.

After the internal planning of the reallocation of the ReconfigurationMaster (here denoted by the function call optimize()) the execution of the reconfiguration is initiated. The ReconfigurationBlock at the destination ECU, where the AudioMaster should be reallocated to, is advised to transfer and activate (defined by the parameter TargetState) the FBlock. Within the simulation a reinstantiation of the

FBlock at the target ECU is carried out. When the reallocation at the source ECU starts, where the AudioMaster is initially located at, the AudioMaster FBlock is deactivated and becomes invalid. The NetworkMaster is informed about the invalid FBlock and broadcasts the information as defined in the MOST standard so that all communication partners of the AudioMaster are aware of its current *invalidity*. Afterwards the transfer of the AudioMaster with its context (characterized by the parameter Type) is performed and the new location of the FBlock is announced by the NetworkMaster.

The described use case with the reallocation of the AudioMaster has been successfully simulated within a MOST network. The implementation and evaluation proves that our new functionalities for self-organization work well in a realistic scenario in the automotive infotainment domain. It also shows that an evolution of present statically designed automotive electronic systems to self-organized systems is feasible.

6 Conclusion

We have presented a potential evolution of automotive electronic systems to self-organizing systems. Thereby, we have identified several challenges of modern automotive systems which need to be addressed in order to achieve self-organization. As an example for an application domain we introduced enhancements for realizing a self-organizing vehicle infotainment network. Those extensions were implemented and validated in a case study of a representative infotainment scenario based on the MOST technology. It shows that the enhanced self-organizing functionalities operate in a realistic infotainment scenario. Thus, we show that the evolution of today's static automotive electronic systems towards self-organizing systems is possible.

As we present our first results in this area, future work can be done in implementing the introduced techniques in embedded devices and analyzing the performance. Also we intend to explore improved algorithms for the planning and execution of self-organization in the infotainment scenario.

References

1. Broy, M.: Challenges in Automotive Software Engineering. In: Proceedings of the 28th International Conference on Software Engineering, pp. 33–42 (2006)
2. De Wolf, T., Holvoet, T.: Designing Self-Organising Emergent Systems based on Information Flows and Feedback-loops. In: First International Conference on Self-Adaptive and Self-Organizing Systems, pp. 295–298 (2007)
3. Müller-Schloer, C.: Organic Computing - On the Feasibility of Controlled Emergence. In: CODES+ISSS 2004: Proceedings of the international conference on Hardware/Software Codesign and System Synthesis, pp. 2–5 (2004)
4. Nagpal, R., Zambonelli, F., Sirer, E., Chaouchi, H., Smirnov, M.: Interdisciplinary Research: Roles for Self-Organization. IEEE Intelligent Systems 21(2), 50–58 (2006)
5. Seebach, H., Ortmeier, F., Reif, W.: Design and Construction of Organic Computing Systems. In: IEEE Congress on Evolutionary Computation, CEC 2007, September 2007, pp. 4215–4221 (2007)

6. Parashar, M., Hariri, S.: Autonomic Computing: An Overview. In: Banâtre, J.-P., Fradet, P., Giavitto, J.-L., Michel, O. (eds.) UPP 2004. LNCS, vol. 3566, pp. 257–269. Springer, Heidelberg (2005)
7. Würtz, R.P.: Organic Computing (Understanding Complex Systems). Springer, Heidelberg (2008)
8. Robertson, P., Shrobe, H.E., Laddaga, R.: Self-Adaptive Software. In: Robertson, P., Shrobe, H.E., Laddaga, R. (eds.) IWSAS 2000. LNCS, vol. 1936. Springer, Heidelberg (2001)
9. Urmson, C., Whittaker, W.: Self-Driving Cars and the Urban Challenge. IEEE Intelligent Systems 23(2), 66–68 (2008)
10. Automotive Open System Architecture (AUTOSAR), <http://www.autosar.org>
11. Trumler, W., Helbig, M., Pietzowski, A., Satzger, B., Ungerer, T.: Self-configuration and Self-healing in AUTOSAR. In: 14th Asia Pacific Automotive Engineering Conference (APAC-14) (August 2007)
12. Dinkel, M.: A Novel IT-Architecture for Self-Management in Distributed Embedded Systems. WiKu (2008)
13. Dynamically Self-Configuring Automotive Systems, <https://www.dyscas.org>
14. Teich, J., Haubelt, C., Koch, D., Streichert, T.: Concepts for Self-Adaptive Automotive Control Architectures. In: Workshop Future Trends in Automotive Electronics and Tool Integration, DATE 2006, Munich, Germany (March 2006)
15. Cakar, E., Mnif, M., Müller-Schloer, C., Richter, U., Schmeck, H.: Towards a Quantitative Notion of Self-Organisation. In: IEEE Congress on Evolutionary Computation, September 2007, pp. 4222–4229 (2007)
16. Rammig, F.: Engineering Self-Coordinating Real-Time Systems. In: 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, May 2007, pp. 21–28 (2007)
17. Kramer, J., Magee, J.: The evolving philosophers problem: dynamic change management. IEEE Transactions on Software Engineering 16(11), 1293–1306 (1990)

Analyzing the Behavior of an Artificial Hormone System for Task Allocation

Uwe Brinkschulte and Alexander von Renteln

Chair of Embedded Systems, Institute for Computer Science, Goethe Universitaet Frankfurt,
Robert-Mayer-Str. 11-15, 60325 Frankfurt am Main, Germany
{brinks, renteln}@es.cs.uni-frankfurt.de

Abstract. In this article we present a detailed theoretical analysis of the behavior of our artificial hormone system. The artificial hormone system (AHS) is part of an organic middleware for mapping tasks on an heterogeneous grid of processing elements. The AHS works completely decentral - each processing cell decides for itself if it is best suited for a task and interacts with the other processing cells via "hormone" messages. As there is no central element the stability of the system can not be controlled by a single processing element, instead the hormone values have to be chosen carefully to guarantee system stability. We will present upper and lower bounds which have to be met to guarantee system stability.

1 Introduction

Today's computational systems are growing increasingly complex. They are build from large numbers of heterogeneous processing elements with highly dynamic interaction. Middleware is a common layer in such distributed systems, managing the cooperation of tasks on the processing elements and hiding the distribution from the application. It is responsible for seamless task interaction on distributed hardware. To handle the complexity of today's – furthermore tomorrow's – distributed systems, self-organization techniques are necessary. The idea to autonomously achieve this desired behavior is introduced in several papers [8][13][10]. Such a system should be able to find a suitable initial configuration by itself, to adapt or optimize itself to changing environmental and internal conditions, to heal itself in case of system failures or to protect itself against attacks.

Middleware is well-suited to realize such self-X features (self-configuration, self-optimization, self-healing) by autonomously controlling and adapting task allocation. Especially for self-healing it is important that task allocation is decentralized to avoid single points of failure. As shown in previous papers [3][4][1] the task distribution done by our artificial hormone system (AHS) works well if a suitable configuration of hormone values is chosen. If the values are chosen badly it can also run out of bounce and get unstable. Due to this we started an investigation to find bounds for the hormone values in which the AHS will work stable and predictable.

The term "artificial hormone system" was chosen, because our approach was highly inspired by the hormone system of higher mammals. There are several properties analog respectively comparable between the hormone system in biology and our technical system. However, it has to be stated that our "artificial hormone system" is not a copy of

the biological hormone system, but is rather inspired by nature and its strategies. In biology, hormones are chemical objects transmitted via chemical processes and reactions. In our approach, the messengers are bits and bytes transferred via communication links. However, the effects and principles are similar, hence we dubbed the messengers in our approach "hormones" as well.

The paper is structured as follows: we will shortly present our approach for decentralized task mapping on heterogeneous processing elements and the concept of the AHS in Section 2. In Section 3 we will then present our theoretical analysis of the system stability of the AHS. Section 4 will examine the limitations of single processing elements. Section 5 contains the related work. The paper finishes with a conclusion and an outlook in Section 6.

2 The Artificial Hormone System

For task allocation, three types of hormones are used:

Eager value: This hormone determines, how suited a processing element (PE) is to execute a task. The higher the hormonal value the better the ability of the PE to execute the task.

Suppressor: A suppressor weakens the possibility of an execution of a task on a PE. Suppressors are subtracted from eager values. Suppressors are e.g. used to prevent duplicate task allocation or to indicate a deteriorating PE state.

Accelerator: An accelerator favors the execution of a task on a PE. Accelerators are added to eager values. The accelerators can be used to cluster cooperating tasks in the neighborhood or to indicate an improved PE state.

Figure 1 sketches the basic control loop used to assign a task T_i to a processing element. This closed control loop is executed for every task on every processing element. It determines based on the level of the three hormone types, if a task T_i is executed on a processing element PE_γ or not. The local static eager value $E_{i\gamma}$ indicates how suited a PE_γ is to execute a task T_i . From this value, all suppressors $S^{i\gamma}$ received for task T_i on PE_γ are subtracted, and all accelerators received for task T_i on PE_γ are added. The result of this calculation is a modified eager value $Em_{i\gamma}$ for task T_i on PE_γ . The modified eager value is sent to all other PEs by the middleware in the system and compared to the modified eager values $Em^{i\gamma}$ received from all other PEs for this task. Is $Em_{i\gamma}$ greater than all received eager values $Em^{i\gamma}$, task T_i will be acquired by PE_γ (in case of equality, a second criterion, e.g. the position of a PE in the grid, is used to get an unambiguous decision). Now, task T_i on PE_γ sends suppressors $S_{i\gamma}$ to all other PEs to prevent duplicate task allocation. Accelerators $A_{i\gamma}$ are sent to neighbored PEs to favor the clustering of cooperating tasks. This procedure is repeated periodically.

It should be emphasized at this point that the strength of the different types of hormones is initially set by the applicants who want to influence the task allocation. More detailed information can be found in earlier published papers given in the references [3][4].

The described approach is completely decentralized, each PE is responsible for its own tasks, the communication to other PEs is realized by a unified hormone concept. This is achieved by employing several self-X properties:

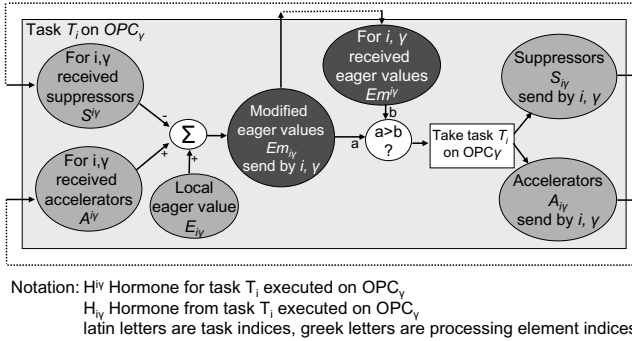


Fig. 1. Hormon based control loop

- The approach is **self-organizing**, because no external influence controls the task allocation.
- It is **self-configuring**, an initial task allocation is found by exchanging hormones. The self-configuration is finished as soon as all modified eager values become zero meaning no more tasks wants to be taken. This is done by sending suppressors which have to be chosen strong enough to inhibit an infinite task assignment.
- The **self-optimization** is done by offering tasks again for re-allocation. The point of time for such an offer is determined by the task respectively the PE itself. It can be done periodically or at a point of time where the task or the PE is idle. In this context, it is simple to handle the entrance of a new task in the system: At first, all processing elements have to be informed about their hormone values for the new task. Then, the task is allocated as described for self-optimization.
- The approach is **self-healing**: in case of a task or PE failure all related hormones are no longer sent, especially the suppressors. This results in an automatic reassignment of the task to the same PE (if it is still active) or another PE.

In addition, the self-configuration is **real-time** capable. There are tight upper time bounds for self-configuration that are presented in detail in [3][4].

3 Stability Analysis of the AHS

By looking at the AHS one can see a variety of parameters and possible configurations that can influence the outcome and the stability of the task mapping. First, there are the hormones themselves: the eager values, the accelerators, and the suppressors. Apart from the already mentioned hormones above, there can be additional local types of accelerators and suppressors for system monitoring, load indication, etc.. Second, there are the degrees of relationship of the tasks. The whole scenario also depends of course on the number of (different) tasks and the number of processing elements.

To be able to determine bounds we start with a very strict configuration with only a few variable parameters and will drop these limitations further and further to get to a generalized approach.

3.1 Suppressors and Eager Values, But No Accelerators

The first configurations are - as already mentioned above - very strict. We only allow suppressors and eager values but no accelerators. With these limitations the AHS will be able to map the tasks but will not regard the relationship between the tasks.

In the first most restricted subconfiguration, the eager values and suppressors have the same value for a task on all nodes. This means all processing elements of the AHS would be equally suited for each task. This can be represented mathematically as follows:

$$eagervalue = ev, \quad suppressor = sup$$

The AHS allocates a task as long as the sum of the suppressors is less than the eager value of the task, i.e. the following inequality is true:

$$n_i * sup < ev \quad (\text{with } n_i: \text{ number of allocations of task } T_i)$$

With this formula it now can be calculated how often a task is allocated:

$$n_i = \left\lfloor \frac{ev}{sup} \right\rfloor$$

We now drop the constraint that all eager values have the same value. The suppressors still have the same value for a task on all nodes but the eager values can be different. We now have the following:

$$eagervalue = ev_{\min, \dots, \max}, \quad suppressor = sup$$

As long as the biggest eager value of the task is not compensated by suppressors the task will be allocated.

Example:

	1. allocation	2. allocation	3. allocation
$ev_1 = 5, sup = 2$	3	1	-1
$ev_2 = 3, sup = 2$	1	-1	-3
$ev_3 = 2, sup = 2$	0	-2	-4

⇒ Task will be allocated three times.

A task T_i will be allocated as long as the following inequalities is true:

$$n_i * sup < ev_{max} \quad (\text{with } n_i \text{ number of allocations for task } T_i)$$

With this formula it can be calculated how often a task T_i will be allocated:

$$n_i = \left\lfloor \frac{ev_{max}}{sup} \right\rfloor$$

Now we allow the suppressors as well as the eager values to be variable. So the suppressors and eager values can have different values on the different nodes:

$$eagervalue = ev_{\min, \dots, \max}, \quad suppressor = sup_{\min, \dots, \max}$$

An upper and lower bound for the number of allocations of a task T_i can now be derived: The upper bound results from a scenario where the task will be allocated on a node with the smallest suppressor. This means allocations will be done as long as the following is true:

$$n_{\max_i} * sup_{\min} < ev_{\max} \quad (1)$$

The lower bound results from a scenario where the task will be allocated on the node with the biggest suppressor meaning tasks will be allocated as long as the following is true:

$$n_{\min_i} * sup_{\max} < ev_{\max} \quad (2)$$

For the number of allocations of a task T_i the following applies:

$$\left\lceil \frac{ev_{\max}}{sup_{\max}} \right\rceil \leq n_i \leq \left\lfloor \frac{ev_{\max}}{sup_{\min}} \right\rfloor$$

As up to now we did not allow any additional local suppressors (e.g. for system monitoring, load indication, etc.). These local suppressors enable the AHS to include local properties of the processing element into the calculation of the processing cell's suitability for a task. So we will now allow that there can be additional local suppressors for a task on the nodes:

$$\begin{aligned} eagervalues &= ev_{\min}, \dots, ev_{\max}, \quad suppressors = sup_{\min}, \dots, sup_{\max}, \\ local\ suppressors &= lsup_{\min}, \dots, lsup_{\max} \end{aligned}$$

The local suppressors are added to the global suppressors. Therefore, it's also possible to specify an upper and lower bound. To get an upper bound of the tasks that will be allocated we simply add the minimum of the local suppressors. Tasks will be allocated as long as the following is true:

$$n_{\max_i} sup_{\min} + lsup_{\min} < ev_{\max}$$

The lower bound can be calculated to:

$$n_{\min_i} sup_{\max} + lsup_{\max} < ev_{\max}$$

To sum up Subsection 3.1 the following can be noticed concerning stability: In all examined cases it is possible to give an upper bound as long as the following is true:

Stability criteria (3.7): If $sup_{\min} > 0$ and $sup_{\max} > 0$:

$$\left\lceil \frac{ev_{\max} - lsup_{\max}}{sup_{\max}} \right\rceil \leq n_i \leq \left\lfloor \frac{ev_{\max} - lsup_{\min}}{sup_{\min}} \right\rfloor^1$$

¹ if the left or right hand term of the inequality gets below 0, it is to be set to 0.

⇒ With this condition the AHS (without using accelerators) is always stable.

3.2 Suppressors, Accelerators and Eager Values

In Subsection 3.1 we limited the scenarios to configurations without accelerator hormones. This constraint will now be dropped.

We will again start with a configuration of the AHS where all hormones have equal values i.e. eager values, suppressors and accelerators have the same value on all nodes:

$$eagervalues = ev, \quad suppressors = sup, \quad accelerators = acc$$

While the suppressor is sent per each taken task, accelerators are sent from each taken task to all related (cooperating) tasks. On this condition a task T_i will be allocated as long as the following is true:

$$n_i * sup < ev + \sum_{j=1}^{a_i} acc * n_j \quad \square$$

An allocated task sends accelerators to all related tasks. This works two-ways, i.e. if task T_i and task T_j are related then T_i will send accelerators to T_j when it is allocated and vice versa. For a group of v related tasks the following system of inequality will result:

$$\begin{aligned} n_1 * sup &< ev + acc * \sum_{j=2, \dots, v} n_j \\ n_2 * sup &< ev + acc * \sum_{\substack{j=1, \dots, v \\ j \neq 2}} n_j \\ &\vdots \\ n_v * sup &< ev + acc * \sum_{j=1, \dots, v-1} n_j \end{aligned}$$

This can be converted to:

$$\begin{aligned} n_1 * sup &< ev + acc * \sum_{j=1, \dots, v} n_j - acc * n_1 \\ n_2 * sup &< ev + acc * \sum_{j=1, \dots, v} n_j - acc * n_2 \\ &\vdots \\ n_v * sup &< ev + acc * \sum_{j=1, \dots, v} n_j - acc * n_j \end{aligned}$$

To calculate the number of possible allocations, we will first solve the corresponding system of equations and afterwards analyze the meaning of this solution for the system of inequalities.

¹ Where a_i is the number of tasks, that send accelerators to task T_i .

$$\begin{aligned}
n_1 * sup &= ev + acc * \sum_{j=1, \dots, v} n_j - acc * n_1 \\
&\vdots \\
n_v * sup &= ev + acc * \sum_{j=1, \dots, v} n_j - acc * n_j
\end{aligned}$$

This results to:

$$\begin{aligned}
n_1 * (sup + acc) &= ev + acc * \sum_{j=1, \dots, v} n_j \\
&\vdots \\
n_v * (sup + acc) &= ev + acc * \sum_{j=1, \dots, v} n_j
\end{aligned}$$

By subtracting any two equations from this system we will receive:

$$\begin{aligned}
n_k * (sup + acc) - n_l * (sup + acc) &= \\
ev + acc * \sum_{j=1, \dots, v} n_j - \left(ev + acc * \sum_{j=1, \dots, v} n_j \right) &= \\
\Rightarrow n_k * (sup + acc) - n_l * (sup + acc) &= 0 \\
\Rightarrow n_k &= n_l \\
\Rightarrow \text{all } v \text{ equations have the same solution} &
\end{aligned}$$

Hence for each task T_i from the set of v related tasks the following applies:

$$\begin{aligned}
n_i * (sup + acc) &= ev + acc * \sum_{1, \dots, v} n_i = ev + acc * v * n_i \\
n_i * sup &= ev + acc * (v - 1) * n_i \tag{3} \\
n_i * (sup - acc * (v - 1)) &= ev \\
n_i &= \frac{ev}{sup - (v - 1) * acc}
\end{aligned}$$

What this means for the solution of the original system of inequalities can be shown graphically with equation [3](#):

The left and the right part of equation [3](#) represent two straight lines. While the left part describes the influence of the suppressors, the right part describes the influence of the accelerators and eager values. The solution of the system of equations is the intersection of the two straight lines where the effect of the suppressors, accelerators, and eager values even out.

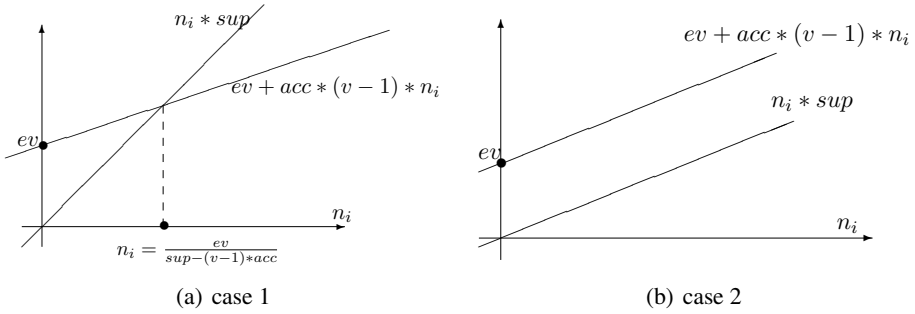


Fig. 2. Graphical interpretation of Equation 3 (part 1)

Three cases can be differentiated:

Case 1: (illustrated in Figure 2(a)) $sup > (v - 1) * acc$

There is a positive solution. On the left side of this solution the eager values and the accelerators are stronger i.e. a task T_i will be further allocated. On the right side of the solution the suppressors are stronger hence there will be no further allocations.

⇒ The task T_i will be allocated exactly $\left\lceil \frac{ev}{sup - (v - 1) * acc} \right\rceil$ times.

Case 2: (illustrated in Figure 2(b)) $sup = (v - 1) * acc$

Both straight lines are parallel hence there is no intersection. This means that for all n_i the eager values and accelerators outperform the suppressors. Task T_i will be allocated infinitely.

Case 3: (illustrated in Figure 3(a)) $sup < (v - 1) * acc$

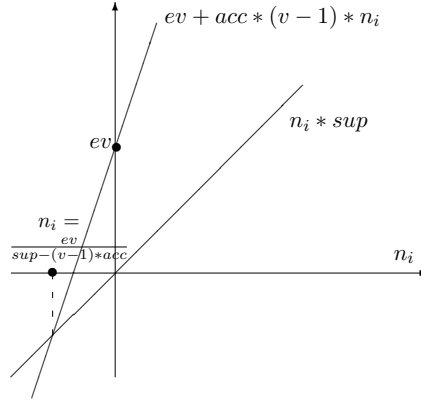
The solution is negative. For all positive n_i the eager values and accelerators outperform the suppressors. As in the case above, here the task T_i will also be allocated infinitely.

To put all the cases together this means the following for the stability of the system:

Stability criteria (3.2): If $sup > (v - 1) * acc$:

$$n_i = \left\lceil \frac{ev}{sup - (v - 1) * acc} \right\rceil$$

All v related tasks will be allocated a finite number of times hence the system is stable. In contrast, if the stability constraint above will not be met, the AHS will try to allocate related tasks infinitely and therefore be unstable.



(a) case 3

Fig. 3. Graphical interpretation of the equation [3](#) (part 2)

3.3 Different Values for Eager Values, Suppressors and Accelerators

In Subsection [3.2](#) we allowed accelerators, suppressors, and eager values but limited these to being equal for a task on all processing cells. Now we drop this limitation. For a set of v related task we now have:

$$\begin{aligned} \text{eager values} &= ev_{\min}, \dots, ev_{\max}, & \text{suppressors} &= sup_{\min}, \dots, sup_{\max}, \\ \text{accelerators} &= acc_{\min}, \dots, acc_{\max} \end{aligned}$$

As before, we can also determine an upper and lower bound for the number of allocated tasks: This can be done in the same way as for Equations [1](#) and [2](#) in Subsection [3.2](#). The maximal eager value combined with upper - respectively lower - bound determines the number of allocated tasks. The AHS will allocate tasks as long as:

$$\text{upper bound:} \quad n_{\max_i} * sup_{\min} < ev_{\max} + \sum_{j=1, \dots, a} acc_{\max} * n_{\max_j}$$

$$\text{lower bound:} \quad n_{\min_i} * sup_{\max} < ev_{\max} + \sum_{j=1, \dots, a} acc_{\min} * n_{\min_j} \quad \text{a}$$

For a group of v related tasks we have the following inequalities:

$$\begin{aligned} n_{\max_1} * sup_{\min} &< ev_{\max} + acc_{\max} * \sum_{j=1, \dots, v} n_{\max_j} - acc_{\max} * n_{\max_1} \\ &\vdots \\ n_{\max_v} * sup_{\min} &< ev_{\max} + acc_{\max} * \sum_{j=1, \dots, v} n_{\max_j} - acc_{\max} * n_{\max_v} \end{aligned}$$

and

² Where a is the number of tasks, which send accelerators to tasks T_i .

$$\begin{aligned}
n_{\min_1} * sup_{\max} &< ev_{\max} + acc_{\min} * \sum_{j=1, \dots, v} n_{\min_j} - acc_{\min} * n_{\min_1} \\
&\vdots \qquad \qquad \qquad \vdots \\
n_{\min_v} * sup_{\max} &< ev_{\max} + acc_{\min} * \sum_{j=1, \dots, v} n_{\min_j} - acc_{\min} * n_{\min_v}
\end{aligned}$$

Both related systems of equations for n_{\max_i} and n_{\min_i} have a solution analog to the one in Subsection 3.2:

$$n_{\max_i} = \frac{ev_{\max}}{sup_{\min} - (v-1) * acc_{\max}} \quad \text{and} \quad n_{\min_i} = \frac{ev_{\max}}{sup_{\max} - (v-1) * acc_{\min}}$$

We can now draw the following stability criteria for the AHS (also analog to 3.2):

Stability criteria (3.3): If $sup_{\min} > (v-1) * acc_{\max}^2$:

$$\left[\frac{ev_{\max}}{sup_{\max} - (v-1) * acc_{\min}} \right] \leq n_i \leq \left[\frac{ev_{\max}}{sup_{\min} - (v-1) * acc_{\max}} \right]$$

² From of this $sup_{\min} - (v-1) * acc_{\max} > 0$ and with $sup_{\max} \geq sup_{\min}$ and $acc_{\max} \geq acc_{\min}$ it follows that $sup_{\max} - (v-1) * acc_{\min} > 0$, too

3.4 Additional Local Suppressors and Accelerators

For a group of v related task, let:

$$\begin{aligned}
eagvalues &= ev_{\min}, \dots, ev_{\max}, \\
accelerators &= acc_{\min}, \dots, acc_{\max}, \text{ local acc.} = lacc_{\min}, \dots, lacc_{\max}, \\
suppressors &= sup_{\min}, \dots, sup_{\max}, \text{ local supp.} = lsup_{\min}, \dots, lsup_{\max},
\end{aligned}$$

Local suppressors reduce the eagvalue while local accelerators elevate the eagvalue. Analog to 3.1 and 3.3 the following upper and lower bounds for task allocation can be derived:

upper bound:

$$n_{\max_i} * sup_{\min} < ev_{\max} - lsup_{\min} + lacc_{\max} + sum_{j=1, \dots, a} acc_{\max} * n_{\max_j}$$

lower bound:

$$n_{\min_i} * sup_{\max} < ev_{\max} - lsup_{\max} + lacc_{\min} + sum_{j=1, \dots, a} acc_{\min} * n_{\min_j} \quad \square$$

³ Where a is the number of tasks which send accelerators to the task t_i .

Now, we can continue analog to [3.2](#) und [3.3](#):

Stability criteria [\(3.4\)](#): If $sup_{\min} > (v - 1) * acc_{\max}$:

$$\left\lceil \frac{ev_{\max} + lacc_{\min} - lsup_{\max}}{sup_{\max} - (v - 1) * acc_{\min}} \right\rceil \leq n_i \leq \left\lfloor \frac{ev_{\max} + lacc_{\max} + lsup_{\min}}{sup_{\min} - (v - 1) * acc_{\max}} \right\rfloor$$

Limited sending range of the accelerators. Accelerators only take effect in the nearby neighborhood \Rightarrow in a worst case scenario a task t_i could get allocated outside of the sending range of the accelerators of the related tasks (e.g. due to notably better eager-values) $\Rightarrow acc_{\min} = 0$

An example in Figure [4\(a\)](#) can illustrate this behavior. In this example task t_2 will get allocated on PE 1, as:

$$\text{on PE 1: } ev_{t_2} = 5$$

$$\text{on PE 5: } ev_{t_2} + acc_{t_2} = 2 + 2 = 4$$

After the allocation the following situation results:

$$\text{on PE 1: } ev_{t_2} - sup_{t_2} = 5 - 5 = 0$$

$$\text{on PE 5: } ev_{t_2} + acc_{t_2} - sup_{t_2} = 2 + 2 - 5 = -1$$

\Rightarrow the task t_2 will not be allocated again

$\Rightarrow t_2$ will be allocated exactly one time, the minimal accelerator $acc_{t_2 \min} = 0$ on PE 1

$$\Rightarrow n_{1 \min} = \left\lceil \frac{5}{5-1*0} \right\rceil = 1$$

(if we would assume $acc_{\min} = 2$ and neglect the limited sending range, we would receive $n_{1 \min} = \left\lceil \frac{5}{5-1*2} \right\rceil = \left\lceil \frac{5}{3} \right\rceil = 2$)

Examination of the load suppressor. The load suppressor is a local suppressor, its strength grows with the number of tasks which have been allocated by a PE. If there are enough processing elements, the upper bound doesn't change, as the load suppressor only applies locally and there are still processing elements left without load. $\Rightarrow lsup_{last} = 0$ This is illustrated in Example [4\(b\)](#).

In regards of the upper bound, the load suppressor can therefore be neglected.

To calculate the lower bound, we have to assume that there aren't enough processing elements available and that all PEs are operating at full load. \Rightarrow lower bound = 0.

However, if we assume that there are enough PEs available, in the lower bound formula ev_{\max} has to simply be replaced by ev_{\min} , as all PEs with ev_{\max} might already be fully occupied.

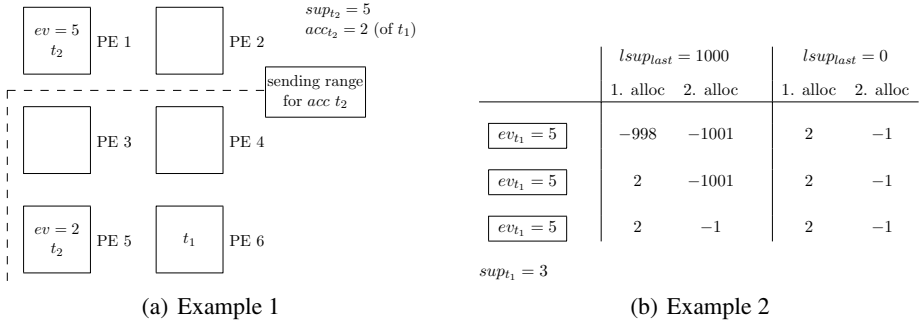


Fig. 4. Examples of the behavior of the AHS

Refinement of the local suppressors und accelerators. Up to now, we used the best case with the biggest combination of eagervalue, local accelerators and suppressors to calculate the upper bound: $ev_{\max} + lacc_{\max} - lsup_{\min}$ and the worst case for the lower bound: $ev_{\max} + lacc_{\min} - lsup_{\max}$. It is now possible to refine the upper and lower bound by using the real values of the biggest possible combination.

$$ev_{\max} + lacc_{\min} - lsup_{\max} \leq \max_{i=1, \dots, v} (ev_i + lacc_i - lsup_i) \leq ev_{\max} + lacc_{\max} - lsup_{\min}$$

From the above results:

$$\left\lceil \frac{\max_{i=1, \dots, v} (ev_i + lacc_i - lsup_i)}{sup_{\max} - (v-1) * acc_{\min}} \right\rceil \leq n_i \leq \left\lfloor \frac{\max_{i=1, \dots, v} (ev_i + lacc_i - lsup_i)}{sup_{\min} - (v-1) * acc_{\max}} \right\rfloor$$

4 Number of Tasks Allocated by One PE

Up to now we analyzed the AHS as a whole. We were able to find boundaries of task allocations in the whole system for different system settings (eagervalue, suppressors, accelerators, etc.). Now we will analyze single processing elements and their behavior.

4.1 Eagervalue, Suppressors, Accelerators and Load Suppressors Are Equal on All PEs

$$eagervalue = ev, \quad suppressor = sup, \quad accelerator = acc, \\ loadsuppressor = lsupl$$

How many tasks can one PE allocate? The load suppressor affects the PE for each allocated task, i.e. with m tasks, the load suppressor is $m * lsupl$, which will be subtracted from all eagervales on this PE. Furthermore the acquisition suppressor will be subtracted for these m tasks. However, these will only affect the eagervalue of the particular task and not all eagervalue like the load suppressor.

To calculate how many tasks will be maximally allocated by a PE, we have to assume the least possible effect of the acquisition suppressors. This is the case, if a node allocates

many different tasks, as the acquisition suppressors will be distributed on many different eagervalues. If a PE applies for b tasks then the acquisition suppressor applies at least $\lfloor \frac{m_1}{b} \rfloor$ on every eagervalue. Furthermore, we assume that every neighbor PE sends this PE an accelerator for every task. Tasks will be allocated as long as the following is true::

$$m_1 * lsupl + \lfloor \frac{m_1}{b} \rfloor * supp < ev + m_1 * k * acc$$

(4)

b : number of tasks that the PE applies for
 k : number of neighbor PEs
 m_1 : maximal number of tasks that the PE can allocate

To calculate how many tasks can be minimally allocated by a PE, we first have to assume the biggest possible number of load suppressors. This is the case if the PE allocates the same task m times, as all acquisition suppressors will effect one eagervalue. Furthermore we will assume, that no accelerators are being sent (hence no related tasks are located on the neighboring PEs).

Tasks will be allocated, as long as:

$$m_2 * lsupl + m_2 * sup < ev \tag{5}$$

We will first solve Equation 4: $m_1 * lsupl + \lfloor \frac{m_1}{b} \rfloor * supp < ev + m_1 * k * acc$

Certainly, the following is true: $\lfloor \frac{m_1}{b} \rfloor > \frac{m_1}{b} - \frac{b-1}{b}$

$$\begin{aligned} \Rightarrow m_1 * lsupl + \frac{m_1}{b} * sup - \frac{b-1}{b} * sup &< ev + m_1 * k * acc \\ m_1 * (lsupl + \frac{sup}{b}) &< ev + m_1 * k * acc + \frac{b-1}{b} * sup \\ m_1 * (lsupl + \frac{sup}{b} - k * acc) &< ev + \frac{b-1}{b} * sup \\ m_1 &< \frac{b*ev+(b-1)*sup}{b*lsupl+sup-b*k*acc} \end{aligned}$$

We can now estimate the maximal number of tasks which will be allocated by the PE by:

$$\text{maximum number of tasks: } m_1 = \left\lfloor \frac{b*ev+(b-1)*sup}{b*(lsupl-k*acc)+sup} \right\rfloor$$

The maximal number of tasks on a PE is in any case limited, if:

$$\begin{aligned} b * (lsupl - k * acc) + sup &\geq 0 \\ b * (lsupl &\geq b * k * acc - sup \\ \Rightarrow lsupl &\geq k * acc - \frac{sup}{b} \end{aligned}$$

Refinement. If $\frac{m_1}{b} - \frac{b-1}{b} < 0$ is true, i.e. $m_1 \leq b - 1$ then $\lfloor \frac{m_1}{b} \rfloor$ can be better estimated by 0 then by the formula above:

$$m_1 = \left\lfloor \frac{ev}{lsupl-k*acc} \right\rfloor \quad \text{if } m_1 \geq b - 1 \text{ and } lsupl > k * acc$$

We now derive Equation 5:

$$\begin{aligned} m_2 * lsupl + m_2 * sup &< ev \\ m_2 * (lsupl + sup) &< ev \\ m_2 &< \frac{ev}{lsupl+sup} \end{aligned}$$

⁴ m_2 : Minimal number of tasks which can be allocated by the PE.

Hence, we can adopt this as the minimal number of tasks that can be allocated by a PE:

$$\text{minimum number of tasks: } m_2 = \left\lceil \frac{ev}{lsupl+sup} \right\rceil$$

This doesn't mean that the PE will allocate this number of tasks. If there are enough other PEs available the PE might stay empty. m_2 rather states the number of tasks i which could be allocated by the PE until the suppressors would not allow any further task allocation.

5 Related Work

Several approaches for clustered task allocation in middleware exist. In [9], the authors present a scheduling algorithm distributing tasks onto a grid. It is implemented in the Xavantes Grid Middleware and arranges the tasks in groups. This approach is completely different from ours because it uses central elements for the grouping: the Group Manager (GM), a Process Manager (PM) and the Activity Managers (AM). Here, the GM is a single point of failure because, if it fails no possibility exists to get group information from this group anymore. Our approach does not apply a central task distribution instance and therefore a single point of failure can not occur.

Another approach is presented in [11]. The authors present two algorithms for task scheduling. The first algorithm, Fast Critical Path (FCP), ensures time constraints are kept. The second one, Fast Load Balancing (FLB), schedules the tasks so that every processor will be used. Using this strategy – especially the last one – it is not guaranteed that related tasks are scheduled nearby each other. In contrast to our approach, these algorithms do not regard failing of processing elements.

[12] presents a load balancing scheme for task allocation based on local workpiles (of PEs) storing the tasks to be executed. The authors propose a load balancing algorithm applied to two PEs to balance their workload. The algorithm is executed with a probability inversely proportional to the length of the workpile of a PE. Although this approach is distributed it does not consider aspects like self-healing or real-time constraints.

Other approaches of load balancing are presented in [2,5,6,7,14]. None of them cover the whole spectrum of self-X properties, task clustering, and real-time conditions like our approach.

6 Conclusion and Further Work

We reintroduced our concept of an artificial hormone system (AHS) to allocate tasks to processing elements within a processor grid. The assignment is completely decentralized and holds self-X features.

We presented a theoretical analysis of the behavior of the AHS. First we restricted the scenario quite strongly and then loosened these restrictions bit by bit. For each scenario we found upper and lower bounds for the number of tasks that will be allocated by the AHS. Furthermore we specified ranges in which the AHS will work stable. Additionally,

we analyzed the limits of single PEs as a step towards being able to set a good system size. It is now easily possible to analyze whether the system will be stable or not.

For the future we plan further investigations to find fitting environments. For example a formula for the minimal number of processing elements could make sure that the system size is chosen correctly.

References

1. Brinkschulte, U., von Renteln, A.: Reliability of an Artificial Hormone System with Self-X Properties. In: *Parallel and Distributed Computing and Systems*, Cambridge, Massachusetts, USA, November 19-21 (2007)
2. Becker, W.: *Dynamische adaptive Lastbalancierung für große, heterogen konkurrierende Anwendungen*. Dissertation, Universität Stuttgart, Fakultät Informatik (December 1995)
3. Brinkschulte, U., Pacher, M., von Renteln, A.: Towards an artificial hormone system for self-organizing real-time task allocation. In: Obermaisser, R., Nah, Y., Puschner, P., Rammig, F.J. (eds.) *SEUS 2007*. LNCS, vol. 4761, pp. 339–347. Springer, Heidelberg (2007)
4. Brinkschulte, U., Pacher, M., von Renteln, A.: *An artificial hormone system for self-organizing real-time task allocation*. Springer, Heidelberg (2008) (to be published)
5. Decker, T., Diekmann, R., Lüling, R., Monien, B.: Universelles dynamisches task-mapping. In: *Konferenzband des PARS'95 Workshops in Stuttgart*. PARS-Mitteilung 14, pp. 122–131 (1995)
6. Finke, J., Passino, K.M., Sparks, A.: Cooperative control via task load balancing for networked uninhabited autonomous vehicles. In: *42nd IEEE Conference on Decision and Control*, Proceedings, vol. 1, pp. 31–36 (2003)
7. Finke, J., Passino, K.M., Sparks, A.: Stable task load balancing strategies for cooperative control of networked autonomous air vehicles. *IEEE Transactions on Control Systems Technology* 14, 789–803 (2006)
8. Horn, P.I.R.: *Autonomic computing manifesto: IBM's perspective on the state of information technology* (October 2001)
9. Bittencourt, L.F., Madeira, E.R.M., Cicerre, L.E., Buzato, L.E.: A path clustering heuristic for scheduling task graphs onto a grid. In: *3rd International Workshop on Middleware for Grid Computing (MGC 2005)*, Grenoble, France (2005)
10. Mueller-Schloer, C., von der Malsburg, C., Wuertz, R.P.: Organic computing. *Informatik Spektrum* 27(4), 332–336 (2004)
11. Radulescu, A., van Gemund, A.J.C.: Fast and effective task scheduling in heterogeneous systems. In: *IEEE Computer - 9th Heterogeneous Computing Workshop*, Cancun, Mexico (2000)
12. Rudolph, L., Slivkin-Allalouf, M., Upfal, E.: A simple load balancing scheme for task allocation in parallel machines. In: *ACM Symposium on Parallel Algorithms and Architectures*, pp. 237–245 (1991)
13. VDE/ITG/GI. *VDE/ITG/GI-Positionspapier Organic Computing: Computer und Systemarchitektur im Jahr 2010* (2003)
14. Xu, C., Lau, F.: Decentralized remapping of data parallel computations with the generalized dimension exchange method. In: *Proceedings of Scalable High-Performance Computing Conference*, pp. 414–421 (1994)

A Software Test Cases Automated Generation Algorithm Based on Immune Principles

Junmin Ye¹, Zemei Zhan^{1,2}, Cong Jin¹, and Qingguo Zhang¹

¹ Department of Computer Science, Central China Normal University,
430079 Wuhan, China

² Computer Science College, Yangtze University, 434023 JingZhou, China
jmye@mail.ccnu.edu.cn

Abstract. Artificial immune operators play an important role in the application of immune system. How to design immune operators to support the test cases generation algorithm based on immune principles deserves our investigation. Therefore, on the basis of artificial immune principles, some immune operators that can be used in software test cases generation are proposed in this paper, on this basis, an algorithm of software test cases automated generation is presented. By analysing our experiment results, the efficiency of the immune operators applied in test case generation method is verified in this paper.

Keywords: Immune System, Immune Operator, Test Case Generation.

1 Introduction

As one of the key technologies in software testing, automated generation of test cases can greatly reduce software testing costs and improve test efficiency. Currently, different ways are researched to generate test cases automatically, such as function oriented test cases automated generation[1-4] and program structure oriented test cases generation[5-8], etc.. Despite the advances in the researches, much remains to be further explored, for example, the optimization process of test cases to avoid random generation of test cases is the focus of our research.

Similar to gene algorithm and other intelligence evolution algorithms, the evolution and optimization of artificial immune algorithm are implemented through designing immune operators, the evolution chain (antibody population→immune selection→cell clone→hyper-mutation→clone suppression→new antibody→new antibody population) in immune response is defined as optimization process in maths [9]. Therefore, the design of immune operators enables the immune algorithm to keep on optimizing the initial test cases generated at random until multi-path test cases are generated. During the process, artificial immune algorithm evaluation mechanism and the quality and density of antibody are used to guide the optimization of test cases, and to avoid generating test cases at random. Hence, the design of the immune operators, test cases automated generation algorithm based on the operators and experimental analysis of the operators performance remain the focus in this research.

The remainder of this paper is organized as follows: Section 2 introduces the research basics, including the working mechanism of the immune system, the work flow of immune algorithm and optimization principles. Section 3 proposes a method to generate test cases on the basis of immune algorithm, including the framework of the algorithm, the selection of parameters and coding strategies and the design of immune operators. Section 4 analyses the experimental results. Section 5 discusses related work. Section 6 is the summarization and discussion of further efforts.

2 Research Foundation

2.1 Biological Immune System

The process of biological immune system discerning antigens and generating antibodies to resist antigens can be divided into primary and secondary immune response. The process of primary immune response is that when antigens invade the first time, Phagocytes extract antigens to form compounds, generating characteristic body of antigens that is discernable to T cell, sending signals to activate T cells and B cells. Activated T cells distinguish the characteristic body of antigens. If the characteristic body is self-antigen, T cells generate signals to resist activation of B cells; otherwise, T cells generate signals to stimulate B cells' activation and help B cells' proliferation and differentiation: T cells excrete a variety of cytokines to prompt B cells to generate B memory cells and antibodies. The process of secondary immune response is that when antigens invade again, as existing B memory cells retain the eigenvalue of the very antigens, antigens rapidly binds with B memory cells to form compounds, stimulating duplication of B cells, and directly generating antibody of strong affinity.

2.2 Basic Flow of Artificial Immune Algorithm and Optimization Principles

For interrelation and mutual support among different artificial immune mechanisms, immune algorithms follow a common basic flow shown in Figure 1.

In Figure 1, antigens are defined in Step1, where problems to be resolved are abstracted as antigens of artificial immune algorithm, a liable solution corresponds to an antibody in immune algorithm; initial antibody population is generated in Step2, where the initial population is generated at random; the affinity between antigens and antibodies is calculated in Step3, this affinity, indicating the matching or similarity between antibodies and antigens and between antibodies, showing the quality of antibodies, plays an important role in guiding antibodies evolution; Step4 is a judgement if the termination conditions are met, if yes then print the current optimal solution and program ends, else go to Step5; density is calculated in Step5 which indicates the diversity and quality of the antibody population; immune operations are conducted in Step6, selecting some optimal antibodies for clone and mutation, reselecting in the combination of the antibodies and the cloned ones to retain those with strong affinity while suppressing the ones with weak affinity, adding new antibodies through population upgrade to ensure the size of the population.

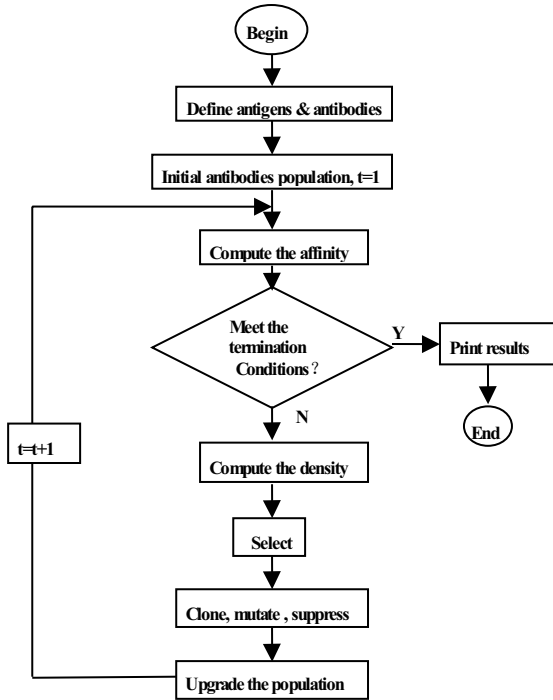


Fig. 1. Basic flow of artificial immune algorithm

3 Test Cases Generation in Light of Artificial Immune Algorithm

Firstly, correspondence should be established between terms of immune system and terms of test cases generation: antigens corresponds to all target paths test cases generated automatically; an antibody to a test case; antibody population to test cases set; affinity to the extent to which a test case in the population satisfies target paths test cases; density corresponds to the percentage of test cases of the same type to the population; stimulation to the quality of the test cases in the final evolution. Then, find the correspondence between optimization process and immune system identifying antigens and antibodies evolving process, that is, abstract the evolution chain in immune response mechanism as a mathematical optimization process^[9].

3.1 The Framework of Test Case Generation Based on Immune Algorithm

There are three steps involved in the test case automated generation based on immune algorithm: (1) program static analysis and instrumentation; (2) deciding on target paths set; (3) the generation of test cases.

Designing immune algorithm according to the process indicated in Figure 1 involves coding antigens and antibodies, binary coding is adopted in this paper. By

program static analysis of source program, we analyse the value range of variables and decide which variables in the test cases set of source program should involve coding; and then, through source program instrumentation, we can obtain information of program executive paths and value of decision conditions.

For the automated generation of multi-path test cases, the target paths set has to be decided on, namely, to generate test cases for what paths? The target path set in this paper is the basic paths set of the program control flow chart.

For automated test case set generation based on artificial immune algorithm, firstly the preliminary test case population is generated randomly by parameter coding rules according to the results of program static analysis, the test cases population is then used as input of the instrumentation program under test to obtain the executive information of the test cases through instrumentation sentences (including program executive paths and the value of decision conditions, etc). In order to get next generation population, we calculate affinity, density and stimulation. From the information, the next generation population is got through immune selection, clone, mutation and population upgrade according to the stimulation of test cases population. If the algorithm's termination condition is satisfied, then terminate the run of the program (or the algorithm) and output the results, otherwise repeat the run of the program (or the algorithm) by using next generation population until the termination condition is met.

3.2 Selecting the Parameters and Using Coding Strategies

Selecting the parameters. The program units under test may include the following three types of parameter variables involved in the coding of test cases: (1) entry parameters, such as formal parameters of function; (2) exterior variables; (3) variables via I/O. According to the rule of "useful parameters coding principle"[10], only the part of variables in program units which are relevant to test paths and to condition expression are coded.

Coding and Decoding strategies. After selecting the parameter variables to be coded through program static analysis, the parameter variables are combined to constitute the test case $X=(x_1, x_2, \dots, x_n)$, where x_i ($1 \leq i \leq n$) denotes the i -th parameter variable of test case. According to the value type and range of the parameter variable, the length of binary coding is set and appropriate coding mapping of each parameter variable is sought:

$$\begin{aligned} f_1(x_1) &\rightarrow c_1 \\ &\dots \\ f_n(x_n) &\rightarrow c_n. \end{aligned}$$

Where x_i is the original parameter variable, c_i is a binary parameter variable, f_i is a function which maps x_i to c_i . The binary coding of variables are combined to get the coded test case $C=(c_1, c_2, \dots, c_n)$, where c_i ($1 \leq i \leq n$) denotes the i -th parameter variable of the test case. The process can be expressed as $f((x_1, \dots, x_n)) = (f_1(x_1), \dots, f_n(x_n)) \rightarrow (c_1, c_2, \dots, c_n)$, that is $f((x_1, \dots, x_n)) \rightarrow (c_1, c_2, \dots, c_n)$.

In order to carry out software testing, we need to use the original test cases, so we must map the binary test cases to the original test cases. We call this process decoding, it can be expressed as follows: $f^{-1}((c_1, \dots, c_n)) = (f^1(x_1), \dots, f^n(x_n)) \rightarrow (x_1, x_2, \dots, x_n)$, that is, $f^{-1}((c_1, \dots, c_n)) \rightarrow (x_1, x_2, \dots, x_n)$.

3.3 Artificial Immune Operators Design

The operators can help select some antibodies for clone and mutation according to their affinity and density. Reselection is based on the clone and the antibodies, the antibodies with weak affinity are suppressed while those with strong affinity retained. The population is maintained by replacing the antibodies with weak affinity with randomly generated new antibodies. Therefore, it is essential to research the design of immune operators used in software test case automated generation. The following research is based on the research work [9] and [11].

Affinity calculation operator. In test case automatic generation, the affinity of each individual within the test cases population reflects the comparability between the individual test case and target path test cases. It is used to guide the evolution of the population test cases to target path test cases. Computing two test cases' comparability can be converted into computing the two paths' comparability, that is, the comparability between the path where the program executes with this test case X_i and the target path.

It is obvious that some statements which have multiple clauses result in multiple executive paths. Since the certain value of the statement's decision conditions corresponds to the certain clause to be executed, the affinity can be computed by using the value of all the clause statements' decision conditions in the program under test. When a statement's decision condition value is certain, the clause to execute is decided, for decision value and clause execution is 1:1 mapping. So clause execution can represent the "True" or "False" of the decision condition. For the sake of convenience, different numbers are used in this paper to represent the executions of different clauses of different statements. The following table shows some specific numbers designated for executions of different clauses of different sentences.

Table 1. *If* statements and the implementation of a clause of the designated values

Executed clause	Designated value
unexecuted clause	-1
True clause	1
False clause	0
True and false clauses	-100

Table 2. *Switch* statements and the implementation of a clause of the designated values

Executed clause	designated value
unexecuted clause	-1
The i th clause	2^i
Multiple clauses to be executed (e.g. the i th, j th, k th clauses)	$2^i 2^j 2^k$

Table 3. For, while, do-while statements and the implementation of a clause of the designated values

Executed clause	designated value
unexecuted clause	-1
Loop	1
No loop	0
Loop and no loop	-100

Every executed clause of the different statements having a designated value means every category of the decision conditions value having a designated value. The decision conditions value P_i corresponding to the test case X_i is a vector composed of these designated values of all decisions in the program. The j -th target path has its corresponding decision conditions value T_j too. P_i and T_j is expressed as follows:

$$P_i = (p_{i1}, p_{i2}, \dots, p_{im})$$

$$T_j = (t_{j1}, t_{j2}, \dots, t_{jm}).$$

Every element in P_i and T_j represents the category of one decision condition value in the program under test. The decision conditions values corresponding to multiple target paths compose a set T whose form is as follows:

$$T = \begin{bmatrix} t_{11}, t_{12}, \dots, t_{1m} \\ t_{21}, t_{22}, \dots, t_{2m} \\ \dots \\ t_{k1}, t_{k2}, \dots, t_{km} \end{bmatrix}.$$

Each line in the matrix T represents the decisions condition value of one target path. The affinity of test case X_i is got by dividing the maximum Hamming Distance between P_i and the vector of the middle line of T by the number of decision conditions of program under test. The affinity $tc_fit(X_i)$ of the test case X_i is defined as follows:

$$tc_fit(X_i) = \frac{D_i}{m}. \quad (1)$$

where $D_i = \max(D_{i1}, D_{i2}, \dots, D_{ik})$.

$$D_{ij} = \sum_{a=1}^m d_{ia}, \begin{cases} d_{ia} = 1 & P_{ia} = t_{ja} \\ d_{ia} = 0 & \text{else} \end{cases}. \quad (2)$$

If the decision conditions values corresponding to the test case are similar to the decision conditions values of one target path, the test case has strong affinity, so it will be chosen into new population.

Density calculation operator. Density reflects the diversity of the test cases population. It is clear that there are too many similar individuals in the test cases population if some test cases' density is high. So the test cases with high density should be suppressed if the

population's diversity is not enough. The density $tc_den(X_i)$ of the test case in the population can be described as follows:

$$tc_den(X_i) = \frac{number_same(X_i)}{N} . \quad (3)$$

$$number_same(X_i) = \sum_{j=1}^k d_{ij} \begin{cases} d_{ij} = 1, P_{i1} = P_{j1}, P_{i2} = P_{j2}, \dots, P_{im} = P_{jm} \\ d_{ij} = 0 \text{ else} \end{cases} . \quad (4)$$

Here, $number_same(X_i)$ stands for the number of test cases with the same decision condition values as X_i . $P_i=(p_{i1}, p_{i2}, \dots, p_{im})$ is the decision condition values corresponding to the test case X_i , letter N for the number of test cases in population. Test cases density is got by dividing the number of test cases of same category by the number of the population. Here the same category refers to the same decision condition value corresponding to test case.

Stimulation calculation operator. The final evaluation of the quality of the test cases is stimulation which takes into account both affinity and density. Obviously the stimulation is proportional to the affinity and is inversely proportional to the density. The stimulation calculation operator $tc_sim(X_i)$ is expressed as follows:

$$tc_sim(X_i) = tc_fit(X_i) - (n-1) \cdot \beta \cdot tc_den(X_i) . \quad (5)$$

where the coefficient before $tc_fit(X_i)$ is 1 and the coefficient before $tc_den(X_i)$ is $(n-1) \cdot \beta$. The number β , which shows how density influences stimulation, is determined by the specific application. Letter n represents the order of the test case to be dealt with in homogeneous test cases with the same decision condition values as x_i . n equals 1 if X_i is the first test case to be dealt with, n equals 2 if X_i is the second test case to be deal with, etc.. The coefficient $(n-1)$ makes stimulations of the same decision condition value of these test cases different, which makes possible immune selection. The earlier test case in homogeneous test cases is processed, the greater its stimulation. The simulation operator can suppress the test cases with high density.

Immune selection operator. The test cases with greater stimulation will be selected into the new population and the test cases with smaller stimulation will be deleted. The Immune selection operator $tc_select(X_i)$ is described as follows:

$$tc_select(X_i) = \begin{cases} 1 & order_sim(X_i) \leq \delta \cdot N \\ 0 & order_sim(X_i) > \delta \cdot N \end{cases} . \quad (6)$$

The coefficient δ is in the area of $(0,1)$. It is the ratio of selected test cases to the population. The function $order_sim(X_i)$ computes the order of X_i in the test cases population by stimulation. The greatest stimulation of x_i is when $order_sim(X_i)$ is 1.

Clone calculation operator. Clone and duplicate test cases are selected by immune selection. Furthermore, a memory paths set must be set up to ensure that the optimal

test cases can enter next generation in test case generation process, all different paths appearing in running process and corresponding test cases are put in the memory paths set, clone $\eta \cdot N$ test cases with the strongest affinity when algorithm clones and duplicates; η is parameter defined in advance, $0 < \eta < 1$, its value represents the ratio of the best test cases cloned to population, ensuring that the best test cases can always be selected to duplicate in test case generation process.

Mutation calculation operator. Mutate cloned test cases. As binary encoding is used in this paper, method to mutate is to change 1 to 0, 0 to 1. Mutation rate of individual test case is inversely proportional to its affinity.

Let test case X_i 's binary coding be $(c_{i1}, c_{i2}, \dots, c_{il})$, letter l represents the length of coding string. Mutation operator $tc_mutate(c_{ik})$ can be depicted as follows:

$$tc_mutate(c_{ik}) = \begin{cases} !c_{ik} & rand() \leq \lambda(1 - tc_fit(X_i)) \\ c_{ik} & rand() > \lambda(1 - tc_fit(X_i)) \end{cases} \quad 1 \leq k \leq l. \quad (7)$$

The $rand()$ is a function to generate random number in range $(0,1)$; λ is parameter defined in advance, $0 < \lambda < 1$, its function is to control the biggest mutation rate. Because the range of affinity calculation operator $tc_fit(X_i)$ is $[0, 1]$, the greater affinity, the smaller $\lambda(1 - tc_fit(X_i))$'s value, the less the possibility of value generated by $rand()$ is smaller than that of $\lambda(1 - tc_fit(X_i))$, and so the less the possibility of mutation. Therefore, mutation rate of test cases is inversely proportional to affinity.

Termination conditions. The algorithm's termination condition can be set as having generated test cases of all target paths or the evolutionary generations exceeding a given number which is relevant to program's scale and the number of target paths, at the same time, ensuring the overall program running time to be within an acceptable scale.

3.4 A Test Case Generation Algorithm Based on Artificial Immune Operators

Based on the study of section 3.1-3.3, suppose that $Decode(PopBin, VarLength)$ stands for the process of decoding, $RunTestedProgram(PopDec)$ for the process of executing test cases of the program under test after instrumentation, $FindNewCase(PopBin, PopDec, PopPath, PopChoice, MemoryPath, TargetPathNoCase, TargetPathCase)$ for the process of finding test cases of target path in population, $ComputeMemoryFit(MemoryPath, TargetPathNoCase)$ for the process of computing the affinity of memory paths, $ComputeFit(PopPath, MemoryPath)$ for the process of computing the affinity of population individual, $ComputeDen(PopPath)$ for the process of computing density of population of test cases, $ComputeStim(PopFit, PopDensity, PopPath)$ for the process of calculating of stimulation of population of test cases, $SelectClone(PopBin, PopStimulation)$ for the process of clone selection, $Mutate(ClonePop)$ for the process of computing mutation, $NextPopGen(ClonePop)$ for the process of population upgrade. A software test case automated generation algorithm based on immune system is shown in figure 2.

Considering the randomness of the artificial immune algorithm and the extremes in analysing algorithm complexity, the maximum evolution generations possible should

be considered. The time complexity of decoding is $O(Nd)$, where N is the size of population, d is the number of parameters. The time complexity of obtaining executed information of test cases of population is $O(N)$. The time complexity of finding target paths test cases is $O(Nn)$, where n is number of paths of memory path set. The time complexity of computing affinity is $O(Nn+nm)$, where m is the number of target paths. The time complexity of computing density and stimulation is $O(N)$, respectively. The time complexity of computing clone selection is $O(N^2)$. The time complexity of computing mutation is $O(NL)$, where L is the length of the binary of a test case. The time complexity of computing upgrade population is $O(N)$.

```

algorithm: generate software test cases in immune operators
input: TargetPathNoCase, EvolveCount;
output: TargetPathCase
Begin
    t=1;          /* t is initial evolution generation*/
    TargetPathCase= ∅; /* initialize TargetPathCase*/

    MemoryPath= ∅; /* initialize MemoryPath */
    Initial(PopBin); /* initialize population of test cases*/

    while(t<10000 and TargetRoadNoCase≠ ∅ )
    { Decode(PopBin,VarLength);
      RunTestedProgram(PopDec);
      FindNewCase (PopBin, PopDec, PopPath, PopChoice, MemoryPath,
                  TargetPathNoCase, TargetPathCase);

      ComputeMemoryFit (MemoryPath ,TargetPathNoCase);
      ComputeFit (PopPath, MemoryPath);
      ComputeDen (PopPath);
      ComputeStim (PopFit, PopDensity, PopPath);
      SelectClone (PopBin, PopStimulation, MemoryPath);
      Mutation (ClonePop);
      NextPopGen (ClonePop);
      t=t+1;
    }/*while*/
End

```

Fig. 2. The algorithm of Immune operators generating test cases

The time complexity of one evolution operation is the maximum value of the operation time complexity discussed in the above paragraph, because d which is the number of parameter is smaller than L which is the length of the binary of a test case, the time complexity of an evolved generation is $\max\{O(N^2), O(Nn+nm), O(NL)\}$. The time complexity of the whole algorithm is $\max\{O(GN^2), O(GNm+Gnm), O(GNL)\}$, where G is the maximum evolved generations, N is the size of population, n is the number of memory paths, m is the number of target paths, L is the length of the binary of a test case.

4 Experiment and Results Analysis

In this article, the sample program's function is to determine the type of a triangle. At first, the source code is statically analyzed, parameters involved in test cases encoding and the parameters' value range are ascertained, instrumentation is done to source codes in order to obtain program's running information; then, according the program's control flow graph, all paths (4 paths) in graph are set as target paths; finally, the immune system-based tools developed by the authors are run to get experimental data.

4.1 The Impact of the Size of Population N on Experiment Results

The impact coefficient of density to the stimulation degree is $\beta=0.05$, immune selection degree is $\delta=0.6$, clone rate of optimization individual is $\eta=0.2$, the coefficient of mutation operation is $\lambda=0.13$, we compare all evolved generations needed to generate test cases of all target paths when the size of population N is 10 and 15, respectively. In order to prevent peak value impact of a single run, we record the results of 20 successive runs. The experiment results are shown in figure 3, where EG is evolution generation and RN is running number, and EG is gradually increasing.

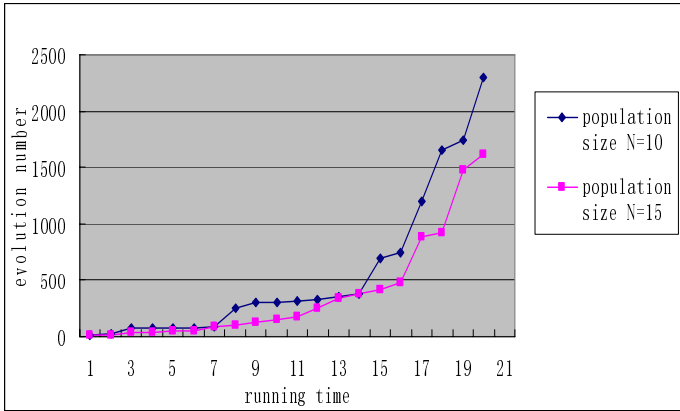


Fig. 3. The impact of the size of population N on experiment results

We can see that the larger the population, the fewer the evolution generations needed to find all test cases. This means the larger the size of the population, the better the diversity, the easier to generate path test cases. On the other hand, the larger the population, the longer the time spent in one evolution operation. So we must consider the two factors in practice.

4.2 The Impact of the Coefficient β of Density to the Degree of Stimulation on Experiment Results

When the size of population is $N=10$, the immune selection degree is $\delta=0.6$, the clone rate of optimization individual is $\eta=0.2$, and the coefficient of mutation operation is $\lambda=0.13$, we can change the impact coefficient β of density to the degree of stimulation and record results of 10 successive runs twice with $\beta=0.05$ and $\beta=0$, respectively. We compare all evolution generations needed to generate test cases target paths. The experiment results are shown in figure 4.

We can see that when $\beta=0$, that is, when the the factor of density is not considered, evolution generations are generally more than those when $\beta=0.05$. Therefore, the guidance of evolution direction of the density operator presented in this paper is effective.

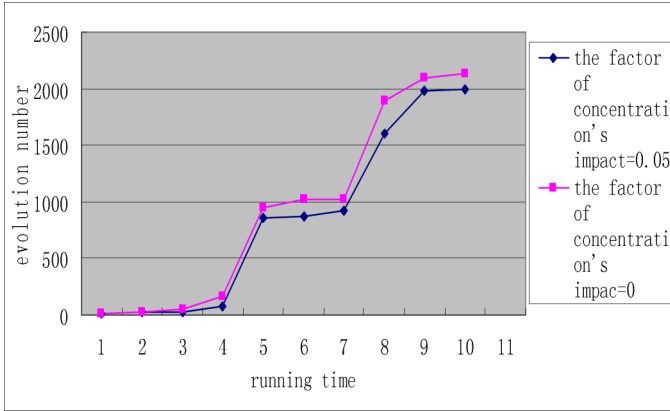


Fig. 4. The impact of the coefficient β of density to the degree of stimulation on experiment results

4.3 The Impact of the Immune Selection Rate δ on Experiment Results

When the size of population is $N=10$, the impact coefficient of density to stimulation degree is $\beta=0.05$, the clone rate of optimization individual is $\eta=0.2$, and the coefficient of mutation operators is $\lambda=0.13$, we can change the immune selection rate δ and record results. The experiment results are shown in figure 5.

We can see from Fig.5 that during cloning part test cases according to stimulation degree on the population, the immune selection rate should be neither too high nor too low. Because they both can slow down the speed to generate test cases of target paths. So the immune selection rate must be suitable.

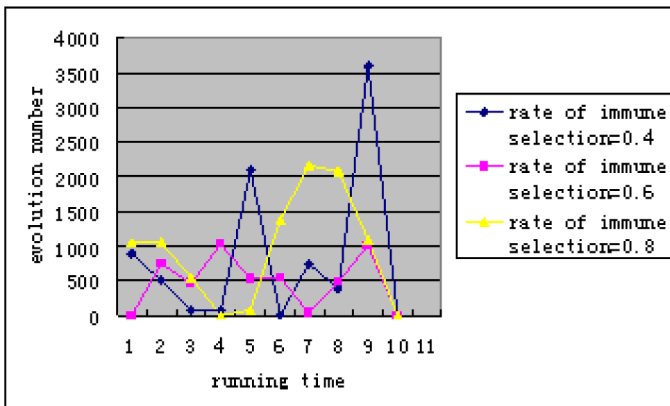


Fig. 5. The impact of immune selection rates δ on the outcome of the experiment

5 Related Work

In recent years, researchers begin to try to apply AI technique to software test and gain lots of results. In [12], the authors present a test data automated generation algorithm based on simulated annealing genetic algorithm. According to the characteristic of automated test data generation, the algorithm combines simulated annealing algorithm which has advantage of local search and genetic algorithm which has advantage of global search. In [13], the authors present a method that can improve generative ability of test data by using genetic algorithm in the process of testing. In [14], the authors present an improved immune algorithm which combines the genetic algorithm with immune algorithm, and can generate component software test cases . Different from those efforts, a method which makes use of immune operators to generate test cases is explored in this paper, which can be described as searching an efficient input data in the input data area and can fall into the category of optimization. For the multi-paths test cases generation, it can be regarded as continual optimization of test cases which are initially generated at random. Because immune algorithm has good optimization ability and can be applied in generating test cases. Our experiments indicate that when the size of population is determinate, this method needs fewer evolution generations than the genetic algorithm does.

6 Conclusions

By utilizing optimization ability of artificial immune algorithm, those test cases initially generated at random are continually optimized till test cases corresponding to the target paths are found. In this paper, the design of immune operators in test case generation is researched on the basis of immune mechanism, and experimental results are discussed. Experimental results manifest the validity of the immune operators design proposed in this paper and the efficiency of the algorithm in generating target path test case.

Acknowledgments. This research has been supported by National Natural Science Foundation (60673118), State Key Laboratory of Software Engineering (SKLSE20080705), Hubei Province Natural Science Foundation (2008CDB349) and Central China Normal University Natural Science Foundation.

References

1. Tsai, W.T., Volovik, D., Tkeefe, T.F.: Automated Test Case Generation for Programs Specified by Relational Algebra Queries. *IEEE Transactions on Software Engineering* 16(3), 316–324 (1990)
2. Weyuker, E.J., Goradia, T., Singh, A.: Automatically Generating Test Case from a Boolean Specification. *IEEE Transactions on Software Engineering* 20(5), 353–363 (1994)
3. PAV Hall. Relationship between Specification and Testing. *Information and Software Technology* 33(1), 47–52 (1991)

4. Hua, L.Y., Yao, M.F., Gong, C.H.: Automated Test Case Generation From Z Specification. *Chinese Journal of Computers* 22(5), 963–969 (1999)
5. Bird, D., Munoz, C.: Automatic generation of random self-checking test cases. *IBM System J.* 22(3), 229–245 (1983)
6. Ramamoorthy, C.V., Ho, S., Chen, W.: On the automated generation of program test data. *IEEE Trans. Software Eng.* SE-2(1), 117–127 (1981)
7. Ramamoorthy, C.V.: On the automated generation of program test data. *IEEE Trans. on Software Eng.* 4, 215–222 (1976)
8. Coward, P.D.: Symbolic execution and testing. *Information and Software Technology* 2, 53–64 (1991)
9. Ning, S.: Artificial Immune Optimization Algorithm and Applications: PhD Thesis. Harbin Institute of Technology, Harbin (2006)
10. Ramanmoorthy, C.V., Ho, S.-B.F., Chen, W.T.: On the automated generation of program test data. *IEEE Trans. on Software Engineering* SE-2(4), 293–300 (1976)
11. JunMin, Y., ZeMei, Z., Wei, D., ZhiChang, Q.: Design of Some Artificial Immune Operators in Software Test Cases Generation. In: *The 2008 International Symposium on Trusted Computing*, pp. 2302–2307. IEEE Press, New York (2008)
12. Bo, F.: Automated Software Test Data Generation Based on Simulated Annealing Genetic Algorithms. *Computer Engineering and Applications*, 85–87 (2005)
13. Zhen, M.: Research on Genetic Algorithm-Based Generating Test Case for Component-Based Software: [Master Degree dissertation]. *Computer Application Technology*, Xi an University of Technology, Xi an (2006)
14. Guangmei, Z., Xiaowei, L., Congying, H.: Automatic Generation of Basis Path Set in Path Test. *Computer Engineering*, 196–197 (2007)

Management without (Detailed) Models

Alva L. Couch¹, Mark Burgess², and Marc Chiarini¹

¹ Computer Science Department

Tufts University

Medford, MA, USA

`couch@cs.tufts.edu, marc.chiarini@tufts.edu`

² Faculty of Engineering

Oslo University College

Oslo, Norway

`mark@iu.hio.no`

Abstract. We present a resource management algorithm based upon guided “walks” within a system state space. Walks are guided via simple predictions of optimum behavior whose accuracy increases as system state approaches a predicted optimum. Optimum behavior is defined as maximizing payoff, which is the difference between value of provided service and cost of providing the service. Feedback between prediction, movement in the state space, and direct observation of behavior allows the algorithm to track optimum payoff, even though there is no detailed model of system behavior. Efficiency of the algorithm is defined as the ratio between observed and optimum payoffs, and can be estimated without reference to a detailed model. We demonstrate by simulation that, under commonly encountered conditions, our algorithm can achieve near-optimal behavior. Our strategy is thus a potentially viable alternative to management based upon closed control loops in many practical situations.

Keywords: autonomic computing, convergent operators, computer immunology, emergent properties, Cfengine.

1 Introduction

Most management paradigms are based upon the possession of a detailed model of assumed behavior and the ability to micro-manage configuration change [1,2,3]. At Hot Autonomic Computing 2008 (HotAC2008), three grand-challenge problems were identified for autonomic computing [4]:

1. Developing more accurate models of system behavior.
2. Ability to compose autonomic control systems and predict behavior of composed systems.
3. Increasing user trust of autonomic systems.

The consensus of the group was that with better models, autonomic control loops become more predictable; with the ability to compose loops, multi-vendor solutions become possible; and that these challenges are linked with the fundamental problem of trust and expectation: the user needs to know what to expect and what to trust.

In this paper we dispute the assumed correlation in prior work between knowledge and outcome, and show that one can (literally) profit by embracing emergent results of highly adaptive management processes.

We approach the challenges above from a different point of view. Before autonomics was proposed, Burgess described a notion of computer immunology [5], based upon the idea that systems can be managed by guiding system states towards desired behavior using what might be considered a thermodynamic (or economic) approach. Autonomic computing can be “approximated” by application of immunological actions [6]; this is the theoretical framework for the management tool Cfengine [7,8]. In an immunological approach, a model of behavior is only indirectly linked with the function of the autonomic system, which can be described as an *emergent property* of an underlying set of processes. In spite of much interest in emergent properties [9,10], few engineers truly believe that emergence can deliver reliable results; this is a part of the trust challenge of autonomic computing. Our experience is that emergent properties can be guaranteed, even within relatively tight tolerances.

We began this work by asking about the kind of autonomic control that would be appropriate to implement in Cfengine. In turn, we asked ourselves how much of a model of system behavior needs to be understood by such an autonomic control system. The answer to this question has turned out to be surprising and counter-intuitive. Rather than “learning” the model of system behavior, one can base a management strategy on remaining flexible and highly reactive to observed changes. This approach is simpler, more adaptive, and more composable than approaches based upon learning models.

2 Cost, Value, and Payoff

Inspired by current work in Business-Driven IT Management (BDIM) [11,12], we took a very basic, high-level view of autonomic control, based upon the objective of balancing concepts of business *cost* and *value*. *Cost* refers to the cost of operating and managing a system, while *value* refers to the value of operating the system. The *payoff* function for operating a system is its value minus its cost, which varies over time. Given some estimate of best payoff, the *efficiency* of a management system is the quotient of the achieved payoff, divided by the best achievable payoff.

Our model of autonomic control is shown in Figure 1. A system consumes resources (R) and exhibits performance (P) subject to outside forces L (load). A *parameter closure* Q [13,14,15,16] controls resource levels based upon interaction with multiple external agents (A) that relay value (V) information to Q . Q knows about the cost $C(R)$ of resources, while the agents know about and describe value

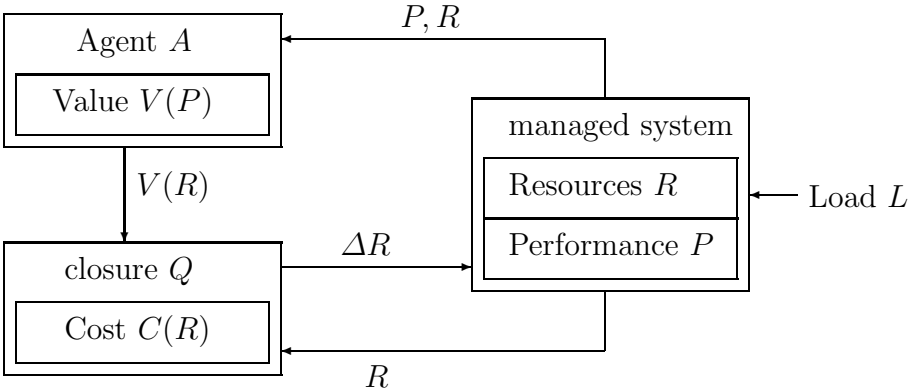


Fig. 1. Our model of autonomic resource control. Q is a resource closure.

$V(P)$ to Q . Q 's job is to manage R to balance value and cost and maximize the net payoff $\pi(P, R) = V(P) - C(R)$.

We incorporate as little information about behavior as possible. In particular, we have no precise knowledge of external load drivers L , other than what can be gleaned from direct observations of P and R . We do not assume that the function $P(R, L)$ (relating resources to performance) is known, nor do we assume any entity in the system can even *measure* L directly other than through its effects on $P(R, L)$. We do not even assume that the transfer function P remains the same function over time; it may vary due to unforeseen forces, e.g., a human being unexpectedly upgrading server hardware.

3 Prior Work

A theoretical formulation of maintaining state was proposed initially by Burgess [5, 17] and refined via several iterations, eventually leading to the concept of promises [18, 19] to represent statements about the intent of autonomous components. The assumptions of this work include that:

1. All agents within a system are autonomous.
2. All cooperation between agents is voluntary.
3. Management occurs in an open world in which unforeseen changes can occur without notice.

Our management paradigm is a natural outgrowth of the assumptions of Cfengine and promise theory, e.g., communications between agents and Q are promises.

¹ To paraphrase an old programming aphorism, Q knows the cost of everything but the value of nothing (like C), while A knows the value of everything but the cost of nothing (like LISP).

In contrast to this, much of autonomic computing is based upon feedback loop controllers [1] which represent so different a management strategy that the two strategies may well be incommensurate:

1. Agents within a system are interdependent.
2. Cooperation is compulsory.
3. Management occurs in a *closed world* in which all (or at least most) achievable system states and management alternatives are known in advance.

One reason it is difficult to compare our approach with what is normally called “autonomic computing” is that the two approaches actually solve different problems. Autonomic computing studies how to manage a controlled environment, such as a data center, while our approach studies emergent properties of management mechanisms in an open world, which includes data centers but also such disparate examples as ad-hoc wireless networks and networks of intelligent appliances. Our mechanisms are best conceptualized as influencing rather than controlling the managed system. While autonomic computing controls the managed system, our approach instead “nudges” the system state toward a more appropriate state over time. Most important, we do not assume that our approach is the sole influence acting on the system.

4 Initial Experiments

Initial simulations of this management paradigm exposed a simple algorithm for maximizing π in the case where C and V are monotonic and invertible functions of R and P , respectively, i.e., $C(R + \Delta R) > C(R)$ for $\Delta R > 0$, and $V(P + \Delta P) > V(P)$ for $\Delta P > 0$. We also assume P is a monotonic and invertible function of R for any constant level L_0 of L , i.e., $P(R + \Delta R, L_0) > P(R, L_0)$ for $\Delta R > 0$. The design and behavior of this algorithm are described in [20].

In the algorithm, agents A send estimates of $\Delta V/\Delta R$ to Q , which combines those estimates with $\Delta C/\Delta R$ to obtain an estimate of $\Delta\pi/\Delta R$. Q then uses this estimate to decide whether to increase or decrease R by some constant increment ΔR . Feedback in the system is expressed in terms of the value estimates $\Delta V/\Delta R$ in response to changes in R .

This algorithm exhibits some surprising behaviors when exercised in the context of a periodically varying (sine-wave) test load L . Very naive estimates of $\Delta V/\Delta R$ (based upon just a few pairs of measurements of V and R) suffice, and more information *degrades* the efficiency of the total algorithm, because errors in estimating $V(P(R, L))$ without knowing about changes in L dominate errors in estimating the function precisely. Thus a very naive estimate of the function $V(R)$ leads to a usable estimate of $\Delta V(P(R))/\Delta R$, and exhibits efficiencies in the high 90’s, even though the estimates of $\Delta V/\Delta R$ appear to be chaotic due to lack of knowledge of L . The algorithm does *not* rely upon any form of machine learning, but instead, relies upon agility in responding to changes in input data, as well as controlled experimentation to determine dynamically evolving alternatives.

It is just as important to note what failed to be useful. Using estimates to determine anything more than the direction in which to change R led to chaos in the managed system, while larger changes in R prevented the agents A from accurately estimating $\Delta\pi/\Delta R$.

5 Analogy with Thermodynamics

The management of low level resources by high level constraints is directly analogous to the study of heat-energy in a hydrodynamic (adaptive) system. Burgess has discussed how computer systems exhibit basic thermodynamic behavior in a number of ways, as they exchange information with their environments [21].

Our management approach is analogous to a ‘heat-pump’ (or refrigerator) for performance. As a system changes (e.g., becomes “cold”) due to the external demand for its energy (performance), we must replenish by supplying the required energy deficit (adding resources). Our aim is to make this process efficient, by pumping just the right amount of resources to meet demand, and not over-allocating resources².

Consider a data center or computer system whose purpose is to deliver a service S at a given rate. S might be measured in transactions per second, for instance³. The computer system is composed of many resources that change at different rates, but the net result is to deliver the service S at a given rate (i.e. a speed of transport). This is analogous to the physics of a hydrodynamic process, in which one has a fluid, composed of atoms flying around at a variety of speeds, and collectively resulting in the transport of fluid at some collective rate v .

At the microscopic level, physics is about collisions and transfer of momentum through interactions, but at the hydrodynamic level, we abstract away these microscopic details and talk about the high level (bulk) properties. We do not need to know about the microscopic details to understand macroscopic behavior, because the two scales are only *weakly coupled*.

“Energy” is a collective scale for quantifying (present or potential) activity. Energy is converted and moved by a transfer of “heat” or “work”, including:

- Q : Heat - an exchange of activity. Heat is modeled as a fluid that can be injected or extracted from a container, just as a service can be delivered. Heat is the service provided by a *thermal reservoir* (e.g. a server array). A flow of heat is a redistribution of service workload.
- U : Internal energy - a store or reservoir of internal capacity. This is a function of the state of a system.
- W : Work - the change incurred by allowing macroscopic motion to take place. This is analogous to computation.

² Though our algorithms can be configured to “over-provision” resources to handle peak load by appropriately modifying concepts of cost and value.

³ In our simulations, we actually measure performance as frequency = 1/rate to make a few calculations more straightforward.

Energy or resources are conserved in a closed system, so

$$U = Q + W$$

i.e., the total internal activity is a result of the heat injected and the work done on the system. Since only differences matter⁴, we often write the closure rule as

$$dU = \tilde{d}Q + \tilde{d}W$$

where the slashed-d's indicate imperfect differentials that cannot be integrated to give estimates of Q or W . Only U is a function of the resources of the system; Q and W are not necessarily functions of resources, but rather, behave like extra, uncontrolled resources themselves.

In our analogy, we must be careful in comparing energy to service. A word like “performance” describes a rate or velocity as a potentially chaotic and dynamic low-level concept, and is not a static, high-level concept like heat. To reason about performance at a high level, we must define how we value that performance or, alternatively, define its energy content. As our high-level energy concept, we define a payoff Lagrangian function $\pi(P, R) = V(P) - C(R)$, where $P = P(R, L)$. Equality in the above is an equilibrium or *quasi-static* property.

To complete the analogy with the hydrodynamic problem:

- Resources R correspond to internal state. So ΔU corresponds to $\Delta C(R)$ – a change in energy state for internal resources.
- Performance $P(R, L)$ contains external effects, driven by changes in R , so ΔP represents work ΔW .
- The payoff π corresponds to heat exchange. We would like to be able to generate or remove heat (depending on the sign), like a pump/refrigerator. So this is our product.

Thus, if C , π , and V were continuous, we might write:

$$dC = \tilde{d}V - \tilde{d}\pi$$

where only C is a true function of R . Since these are in our case step functions, we write $\Delta C = \Delta V - \Delta\pi$, which corresponds to our definition of π .

Note that we are describing changes to a steady state, not the steady exchange of service transactions. $C(R)$ is a function of state, but $V(P)$ is not, since we do not have a closed expression for it. To make a complete deterministic model one would need to relate L to π .

Thermodynamics shows us that we do not need detailed information about this relationship in most cases. In thermodynamics, it is usually sufficient to describe scenarios of constant entropy, which are projections of the external system into our state-space. We thus employ an *equilibrium* approach to bypass the need for information we do not have^[22].

⁴ There is no absolute concept of value, without grounding.

6 Discretizing Payoff

The continuum algorithm described in Section 4 requires monotonic and invertible V , C , and P . It cannot be applied when V or C are not invertible, e.g., when V and C are monotonic *step functions* that map multiple performance values and resource levels to the same value and cost, in like manner to many common Service-Level Agreements (SLAs). Viewed as a function, the simplest possible SLA is a function $V(P)$ that is 0 unless requirements (lower bounds) for P are met, and has some constant payoff v_0 if those requirements are met. In like manner, the cost of a particular physical system configuration is often a step function, e.g., one might allocate a whole rack at a time to a task in a cloud. Of the functions in the original scheme, only $P(R, L)$ remains invertible in R for fixed L : adding resources always improves performance (at least somewhat).

Again, algorithms for managing this system emerged via simulation of several possible options. The new algorithms have little resemblance to the algorithms for the invertible case, but also expose a bit more of the quandaries involved in this form of management.

7 Our Algorithm

The goal of the new algorithm is to estimate which way to change R based upon a limited amount of performance data (expressed as ordered pairs (R_i, P_i)). The goal of changing R is to maximize a payoff function $\pi = V(P) - C(R)$ where $C(R)$ is a cost step-function and $V(P)$ is a value step-function. The main problem in combining these functions into a meaningful estimate of payoff π is that P and R are not *commensurate*, in the sense that there is no function $P(R)$ such that $V(P(R)) = V(R)$ represents value. P is not just a function of R , but also a function of L . If there were such a function $P(R)$, then $\pi(R) = V(R) - C(R)$ would be an estimate of the total payoff function, which we can maximize. This is only true, however, if L is held constant, and then represents an *equilibrium state* for the managed system.

In the new algorithm, each agent A makes an estimate $P_{\text{est}}(R)$ of $P(R)$ from a small number of pairs of samples (R_i, P_i) using linear least-squares approximation. In this paper, the number of samples used for this approximation is $w = 10$. The agent uses this estimate to compute an estimate of the value function $V_{\text{est}}(R) = V(P_{\text{est}}(R))$. This is a matter of transforming only the input axis to the step function V , and the output axis does not change; converting $V(P)$ to $V_{\text{est}}(R)$ only moves the transition boundaries at which the step function V changes value in its domain, and does not affect its range. As P is monotonic and invertible in R , the axis transformation always leaves the transition boundaries in V in the same *order*, so that $V_{\text{est}}(R)$ remains simply increasing like $V(P)$.

Each agent A then forwards its estimate of the function $V(R)$ to Q , which subtracts the function $C(R)$ to estimate the function $\pi(R)$. Relative to the current value of R , Q computes the nearest value R_{target} of R at which the estimate of $\pi(R)$ is maximum. Q increments R in the direction of R_{target} , and then the whole process repeats with new agent estimates of V .

Note that the algorithm behaves predictably even though its estimates are often inaccurate and its decisions are often wrong, so much so that we might characterize the estimates as chaotic⁵. Each individual estimate is a micro-scale event, which can be far from accurate, while overall behavior at the macro scale remains near-optimal. As in the invertible case, the estimates of P (and thus of V) are highly error-prone, in the sense that changes in the hidden variable L are not accounted for in the model. But, for the algorithm to work properly, all that is required is that the estimates are *statistically correct*, or “correct on average”. When L stops changing as rapidly, estimates become more accurate and the management system corrects its prior errors in judgment.

The algorithm also actively compensates for estimation errors. A discards duplicate, older measurements of P for the same R when estimating $P(R)$. Q employs several heuristics, including:

1. Limiting the impact of an erroneous estimate by limiting the resulting change in R to a constant ΔR per time step.
2. Keeping R relatively constant whenever its estimate indicates that R is optimal.
3. Periodically checking (every w steps) that a seemingly optimal resource setting R remains optimal, by perturbing R by one unit ΔR *inside* an optimal zone. This perturbation results in a revised estimate of V .

8 Time-State Diagrams

A crucial difference between the invertible and step-function cases is that “time-state-diagrams” become important to understanding the behavior of the algorithm. The step functions in effect at a given time t_0 (for a given unknown value L_0 of L) determine a global payoff function $\pi(R) = V(P(R, L_0)) - C(R)$ that separates the values of R into a state diagram with specific payoffs. Our estimates $\pi_{\text{est}} = V_{\text{est}}(R) - C(R)$ are also estimates of time-state diagram structure. The best possible payoff for π is approximated by the region with maximum payoff in π_{est} .

One way to exhibit the behavior of the algorithm is via a “time-state diagram” as depicted in Figure 2. Each diagram of the figure has a background that is a theoretical time-state plot for R , based upon a known L that is not measurable by agents A or closure Q . The x-axis of each diagram represents time while the y-axis represents values of R . The horizontal and curved lines on the diagram represent theoretical cutoffs at which step-functions change in value. The horizontal lines represent increases in the cost function C , where cost increases from below to above. The curved lines represent theoretical increases in the value function $V(P(R, L))$ due to changes in L , where value increases from below to above. The curved lines are based upon complete (theoretical) information for L that

⁵ But this is a misconception due to lack of information. It is not truly “chaotic” in a dynamical systems sense, because L is deterministic and determines P , but the fact that we cannot *observe* L makes P seem to behave randomly *to an observer*.

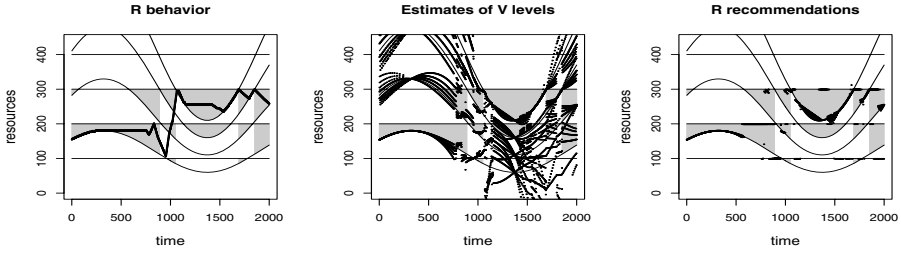


Fig. 2. Attempting to estimate all of the cutoff points for the step-function V leads to seemingly chaotic predictions but generally acceptable tracking

agents A do not possess and must estimate. The curved lines vary in a periodic pattern based upon a sine-wave form for L .

It is best to think of V and C not as functions, but as *sets of constraints* on value and cost. In each of these diagrams, since the horizontal and curved lines represent all transitions in payoff, each delimited region bounded by lines is a zone of constant payoff; a value of R anywhere in a region achieves the same payoff. The theoretical regions of maximum payoff are shaded in gray; these are the target regions for R . In this figure there are several target regions for each point in time, corresponding to several different solutions to obtaining the same overall payoff.

In the figure, the left-hand diagram contains the trajectory for R over time. $\pi = V - C$ is optimal when R lies in a gray region, and non-optimal otherwise. The middle diagram depicts the algorithm's estimates for value level curves, where each circle represents an inference of some cutoff in the value function. These are only approximations and are often in error; the seemingly chaotic changes in these estimates will be useful later. The right-hand diagram depicts the algorithm's estimates of optimal R , one per cycle.

9 Performance Evaluation

In evaluating this and similar algorithms, one cannot rely on traditional techniques for performance evaluation that compare performance to a theoretical best case. While in our simulations the theoretical best case is known, in realistic situations, the theoretical best case is not merely unknown. Given our framework of assumptions, it is also *unknowable*, in the sense that there is no *mechanism* for observing and learning about load L , and thus no mechanism for fully understanding performance $P(R, L)$.

There are subtle benefits to limiting knowledge. Consider a scenario in which a hacker gains access to a computer and starts utilizing resources. In a traditional autonomic control environment, the performance model would be invalidated by this hidden resource drain, while in our case, there is no model to invalidate. Instead, our algorithm would adapt to the new behavioral conditions without

even recognizing that there has been an overall change in state, and try to guarantee performance in the *presence* of the intrusion. Our strategy is not an intrusion-detection tool: it reacts to anomalies *without detecting them*.

Another positive attribute of lack of knowledge is that multiple management algorithms of this kind can be composed without knowledge of one another, e.g., to manage multiple resources. Each algorithm instance will influence L , which will indirectly influence other instances, but the overall effect will be that each algorithm optimizes its own parameters in isolation from the others. Coordination is achieved by mutual observation, and not by communication.

9.1 Observable Performance Problems

While comparing the algorithm to best case behavior is infeasible, several kinds of performance problems are readily observable in a trace of the algorithm's behavior:

1. *Inaccuracy* resulting from inaccurate estimates of V and inappropriate recommendations for changing R .
2. *Lag* between observing a performance problem and reacting via a change in R .
3. *Undershoot* in which R takes time to approach an optimal value (either from above or below).
4. *Overshoot* in which the increment ΔR chosen for R is too large to allow R to remain in a small optimal region.

It is somewhat ironic that an algorithm that does not explicitly consider time can only be analyzed by considering time. The best way to understand the algorithm's behavior is to remember that all of our parameters are functions of time, so that $R = R(t)$ and $L = L(t)$, and thus $P = P(t) = P(R(t), L(t))$. All of the observable performance problems can be characterized in terms of the behavior of P over time, compared with the behavior of R .

Lag in the algorithm is a direct result of any change whatever in L . By the time information is available, performance has been substandard for a measurement cycle already. This kind of error is always present any time the resource level becomes non-optimal.

Undershoot occurs when a new recommendation for R is distant from a current value. In this case, we see a sequence of linear changes in R that may or may not map to actual changes in payoff. When we observe an undershoot in the trace, all we can do is to estimate the *worst-case* scenario in which the undershoot happened for justifiable reasons. It may also be that the current value of R is optimal and the undershoot does not affect the payoff function, or that the undershoot is due to pursuing an erroneous recommendation.

Overshoot describes a condition in which the increment ΔR of R is too large to allow R to settle into a region of maximum payoff. The symptoms of overshoot include rapid oscillation between two adjacent values of R with no settling down. This is due to the algorithm recommending a downturn for the high value and an upturn for the low value.

9.2 Efficiency

To quantify the algorithm's behavior, we must develop a high-level concept of efficiency. The efficiency of our management paradigm during one time step might be characterized as

$$\pi^{\text{observed}} / \pi^{\text{best}}$$

where π^{observed} represents one measurement of observed payoff and π^{best} represents a corresponding (theoretical) maximum payoff. The efficiency of our management paradigm over time might be defined as the ratio of observed payoff to best payoff, e.g., the ratio

$$E^{\text{theoretical}} = (\sum \pi^{\text{observed}}) / (\sum \pi^{\text{best}})$$

where sums are over time. In both cases, the maximum efficiency is 100 percent, which corresponds to a situation in which observed and theoretically best payoffs always agree. This is impossible due to observation lag, and a certain amount of efficiency is thus lost "by design" due to each change in L , even if the management system is functioning perfectly.

In our simulations, we can compute this notion of efficiency, but in practical situations, the theoretical quantity π^{best} is both unknown and unknowable. Efficiency thus cannot be computed directly, because there is no predictive model of payoff, and thus no concept of best performance. One can, however, bound efficiency by estimating best-case behavior and comparing that estimate to observed behavior.

In bounding best-case behavior, the apparent chaos in our estimates V_{est} of V actually helps. On average, the maximum payoff predicted by $\pi^{\text{est}}(R) = V^{\text{est}}(R) - C(R)$ is an *upper bound* for the theoretical maximum payoff $\pi(R) = V(R) - C(R)$, because statistically, *some* of the estimates of V^{est} are accurate. Thus, provided a large-enough sample of successive estimates π_i^{est} ($i = 1, \dots, n$) is considered, the maximum of the upper bounds of the set of estimates is an upper bound on the best-case payoff in the middle of the window:

$$\pi_{n/2}^{\text{best}} \leq \pi^{\text{bound}} = \max_{i=1}^n (\max(\pi_i^{\text{est}}))$$

where the inner max represents the maximum payoff predicted by one estimate function π_i^{est} , while the outer max represents the maximum over the set of estimate functions.

Thus a more practical notion of efficiency is an "observed efficiency bound", defined as

$$E^{\text{bound}} = (\sum_i \pi_i^{\text{observed}}) / (\sum_i \pi_i^{\text{bound}})$$

Since empirically $\pi_i^{\text{bound}} \geq \pi_i^{\text{best}}$, $E^{\text{bound}} \leq E^{\text{theoretical}}$, so it is always a lower bound.

10 Simulation Results

To characterize the behavior of this algorithm, we proceed via examples. We simulated the algorithm using a custom simulator written in the R statistics

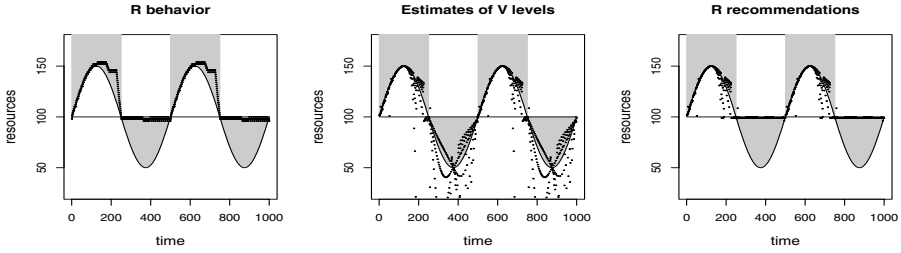


Fig. 3. The simplest management problem consists of one cost change and one value change. In this example, the period of the load function is 500 and ΔR is 2. Theoretical efficiency is 83%, while observed efficiency is 75%.

Table 1. Performance of the two-constraint payoff model. The increment ΔR can be tuned by comparing observed efficiencies of values. The boldface entry corresponds to Figure 3. When observed efficiency is greater than theoretical efficiency, this is due to lack of seemingly chaotic behavior.

period	$\Delta R = 1$	$\Delta R = 2$	$\Delta R = 3$	$\Delta R = 4$	$\Delta R = 5$	$\Delta R = 6$	$\Delta R = 7$	$\Delta R = 8$
100	33.6 34.5	28.9 29.3	45.6 50.7	49.7 43.3	82.6 65.1	73.2 59.6	77.2 61.8	78.5 62.9
200	30.8 34.6	81.9 69.0	74.6 61.6	76.6 62.2	82.6 71.6	86.0 73.2	86.3 74.8	91.3 79.6
300	66.8 77.7	71.9 61.4	81.5 70.5	85.7 73.9	89.5 80.1	92.2 83.0	93.1 83.4	93.8 85.1
400	67.0 75.2	79.9 70.9	86.4 76.8	92.8 83.9	90.6 82.1	95.0 87.1	95.1 87.1	95.3 88.6
500	68.7 60.9	83.1 75.6	90.8 82.2	94.0 85.8	93.8 86.1	95.3 89.1	96.0 89.6	95.9 90.3
600	74.6 64.0	85.1 75.7	93.2 85.1	95.1 88.1	93.2 87.5	95.8 90.4	96.9 91.4	96.3 90.9
700	77.3 71.0	90.8 82.2	95.0 87.3	95.8 89.9	93.9 89.6	96.3 91.4	97.1 92.2	96.6 92.1
800	79.9 73.9	92.6 84.7	94.6 87.7	96.1 90.5	93.8 89.5	96.4 92.3	97.1 93.0	96.8 92.7

language [23]. In our simulation, the hidden theoretical model of performance is $P(R, L) = R/L$, so that P represents frequency response with perfect scalability of resources, e.g., for a web farm. L is a sine wave varying between (unitless) loads with magnitudes 0.5 and 1.5, corresponding to frequency responses between $R/0.5$ and $R/1.5$. We vary period (and thus frequency) between 100-800 time steps per cycle, and ΔR between 1 and 8. This gives us an idea of the interaction between changes in hidden variables and efficiency of the management system.

Inputs to the algorithm include step-functions that we will notate using characteristic ($\chi_{[a,b]}$) notation, where the characteristic function $\chi_{[a,b]}$ is defined by

$$\chi_{[a,b]}(X) = \begin{cases} 0 & \text{if } X \notin [a, b) \\ 1 & \text{if } X \in [a, b) \end{cases}$$

and $[a, b)$ represents the half-open interval from the number a to the number b , not including b .

Our first example is the simplest non-trivial management problem, consisting of one cost transition and one value transition. The cost function is $C(R) = 100\chi_{[100,\infty)}(R)$ while the value function is $V(P) = 200\chi_{[100,\infty)}(P)$. One sample run is shown in Figure 3, whose interpretation is identical to that of Figure 2.

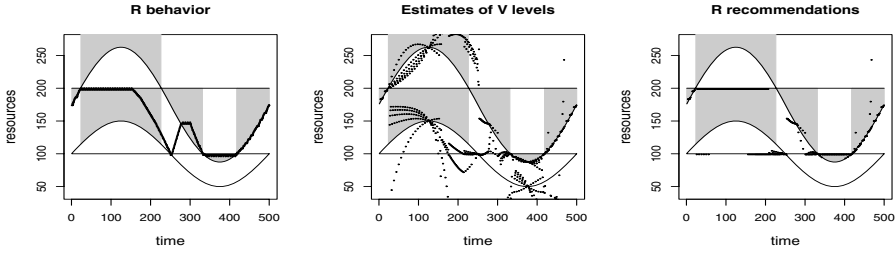


Fig. 4. A slightly more complex management problem with two cost constraints and two value constraints. In this example, the period of the load function is 500 and ΔR is 2. Theoretical efficiency is 74%, while observed efficiency is 65%.

Table 2. Performance of the four-constraint payoff model. The boldface entry corresponds to Figure 4. Observed efficiency is sometimes not commensurate with theoretical efficiency due to periods of non-prediction; those observed efficiencies are omitted.

period	$\Delta R = 1$	$\Delta R = 2$	$\Delta R = 3$	$\Delta R = 4$	$\Delta R = 5$	$\Delta R = 6$	$\Delta R = 7$	$\Delta R = 8$
100	48.5 52.8	87.7 -	62.1 -	80.0 61.8	68.1 54.8	50.2 45.6	51.9 40.8	80.4 56.8
200	77.1 78.1	72.8 61.4	65.1 -	56.5 45.4	74.1 62.2	85.4 65.0	87.8 69.8	88.0 70.9
300	85.3 71.3	57.6 45.3	71.8 60.8	75.1 64.8	86.5 70.9	90.3 76.0	91.7 77.9	92.3 79.6
400	84.1 -	60.4 52.8	87.3 71.8	89.6 76.1	79.1 68.5	91.9 80.5	94.3 83.0	93.6 83.5
500	64.5 -	74.0 64.5	89.3 75.9	82.7 72.9	90.9 80.6	93.1 83.4	95.3 85.7	93.9 86.0
600	61.3 49.1	87.3 73.6	90.8 79.0	85.8 76.1	91.7 82.8	93.8 85.2	95.7 87.8	93.9 86.9
700	51.8 45.0	88.9 75.7	91.7 80.7	94.6 84.6	92.7 84.6	94.7 87.1	96.2 88.6	94.6 88.2
800	63.9 56.8	89.7 78.1	93.0 83.2	89.4 80.7	93.1 85.4	94.5 87.7	96.1 90.2	94.5 88.9

Even in the simplest example, there is a relationship between rate of change of the hidden variable L , and choice of increment ΔR . The relationship between rate of change, choice of increment, and efficiency is shown in Table 1, and demonstrates that one can tune ΔR to ΔL by experimentation.

Our second example is a slightly less trivial management problem with two levels for each of cost and value:

$$C(R) = 100\chi_{[100,200)}(R) + 300\chi_{[200,\infty)}(R)$$

$$V(P) = 100\chi_{[200,400)}(P) + 175\chi_{[400,\infty)}(P)$$

An example run is shown in Figure 4 and performance data comparing periods and increment values is shown in Table 2. Efficiencies are higher because the travel time between desirable states is smaller, but sometimes, the predictor does not predict a best value, due to rapid changes in R that lead to inappropriate predictions for V . If this happens for an extended period, the observed efficiency cannot be compared with theoretical efficiency, because there is no prediction of best-case behavior for the time when predictions are withheld. In that case(which the algorithm detects), observed efficiencies are not comparable with theoretical efficiencies and are thus omitted from the table.

11 Conclusions

We have demonstrated that order can arise from apparent chaos, with a bit of help. Seemingly chaotic underlying approximations of the high-level state diagram inform resource changes that are observably stable; apparent chaos serves as an effective search aid in planning changes that lead to increased payoff. One key to the emergence of order is that estimates of state are employed only to choose the direction of resource change, while recommended changes serve as experiments that – in turn – refine state estimates near the current resource level. This feedback between experiment and recommendation leads to high-level predictability even though low-level estimates remain unpredictable. At the same time, the nature of the low-level estimates enumerates possibilities in a way that allows estimation of a lower bound on overall efficiency. Because estimates are of the payoff *function*, rather than the payoff *value*, global optimization is possible, which was not possible in prior formulations.

Many open issues remain. The multi-dimensional case has yet to be studied, though the approach here has promise by use of multi-dimensional linear-least squares approximations of $P(R)$ and $V(R)$. Another basic issue is that while the value of performance ($V(P)$) is rather well understood, most sites do not have a clear idea of the cost ($C(R)$) of resources, without which a balance cannot be reached. One possible avenue of study is to link this cost with power awareness. Several design issues also require further study. There are many options for implementing various kinds of policies in this setting, and the best options remain unclear. For example, what is the best way to implement an “over-provisioning” policy?

In the usual concept of autonomics, measuring is easy but modeling is hard. We have constructed a situation in which the reverse is true: modeling is easy but precisely computing efficiency is not possible, though it stays near optimum. Our approach allows us to do without detailed modeling, with benefit in a large number of practical situations.

References

1. Hellerstein, J.L., Diao, Y., Parekh, S., Tilbury, D.M.: Feedback Control of Computing Systems. John Wiley & Sons, Chichester (2004)
2. Horn, P.: Autonomic computing: Ibm’s perspective on the state of information technology (October 2001), http://researchweb.watson.ibm.com/autonomic/manifesto/autonomic_computing.pdf (cited April 16, 2009)
3. IBM: An architectural blueprint for autonomic computing (June 2006), http://www-01.ibm.com/software/tivoli/autonomic/pdfs/AC_Blueprint_White_Paper_4th.pdf (cited 16 April 2009)
4. Couch, A.: Summary of the third workshop on hot topics in autonomic computing (hotac iii). *Login: Magazine* 33(5), 112–113 (2008)
5. Burgess, M.: Computer immunology. In: Proceedings of the Twelfth Systems Administration Conference (LISA XII), p. 283. USENIX Association, Berkeley (1998)
6. Burgess, M., Couch, A.: Autonomic computing approximated by fixed-point promises. In: Proceedings of the First IEEE International Workshop on Modeling Autonomic Communication Environments (MACE), pp. 197–222. Multicon Verlag (2006)

7. Burgess, M.: Configurable immunity for evolving human-computer systems. *Science of Computer Programming* 51, 197 (2004)
8. Burgess, M.: A site configuration engine. *Computing Systems* 8(2), 309–337 (1995)
9. Holland, J.H.: *Emergence: From Chaos to Order*. Oxford Univ. Pr. (Sd), Oxford (2000)
10. Johnson, S.: *Emergence: The Connected Lives of Ants, Brains, Cities, and Software*. Scribner (September 2002)
11. Sauve, J., Moura, A., Sampaio, M., Jornada, J., Radziuk, E.: An introductory overview and survey of Business-Driven IT management. In: *The First IEEE/IFIP International Workshop on Business-Driven IT Management (BDIM)*, pp. 1–10 (2006)
12. Moura, A., Sauve, J., Bartolini, C.: Research challenges of Business-Driven IT management. In: *The Second IEEE/IFIP International Workshop on Business-Driven IT Management (BDIM)*, pp. 19–28 (2007)
13. Couch, A., Hart, J., Idhaw, E.G., Kallas, D.: Seeking closure in an open world: A behavioral agent approach to configuration management. In: *LISA 2003: Proceedings of the 17th USENIX Conference on System Administration, Berkeley, CA, USA*, pp. 125–148. USENIX (2003)
14. Schwartzberg, S., Couch, A.: Experience implementing a web service closure. In: *LISA 2004: Proceedings of the 18th USENIX Conference on System Administration, Berkeley, CA, USA*, pp. 213–230. USENIX (2004)
15. Wu, N., Couch, A.: Experience implementing an ip address closure. In: *LISA 2006: Proceedings of the 20th USENIX conference on System administration, Berkeley, CA, USA*, pp. 119–130. USENIX (2006)
16. Couch, A.L., Chiarini, M.: A theory of closure operators. In: Hausheer, D., Schönwälder, J. (eds.) *AIMS 2008*. LNCS, vol. 5127, pp. 162–174. Springer, Heidelberg (2008)
17. Burgess, M.: On the theory of system administration. *Science of Computer Programming* 49, 1 (2003)
18. Burgess, M.: An approach to understanding policy based on autonomy and voluntary cooperation. In: Schönwälder, J., Serrat, J. (eds.) *DSOM 2005*. LNCS, vol. 3775, pp. 97–108. Springer, Heidelberg (2005)
19. Bergstra, J., Burgess, M.: A static theory of promises. Technical report, arXiv:0810.3294v1 (2008)
20. Couch, A.L., Chiarini, M.: Dynamics of resource closure operators. In: Sadre, R., Pras, A. (eds.) *AIMS 2009*. LNCS, vol. 5637, pp. 28–41. Springer, Heidelberg (2009)
21. Burgess, M.: Thermal, non-equilibrium phase space for networked computers. *Physical Review E* 62, 1738 (2000)
22. Burgess, M.: Keynote: The promise of self-adapting equilibrium. In: *Proceedings of the Fifth IEEE International Conference on Autonomic Computing (ICAC)* (June 2008)
23. R Development Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria (2008) ISBN 3-900051-07-0

Formal Development of Self-organising Systems

Graeme Smith^{1,*} and Jeffrey W. Sanders^{2,**}

¹ School of Information Technology and Electrical Engineering
The University of Queensland, Australia

² International Institute for Software Technology
United Nations University, Macao SAR China

Abstract. Attempts to engineer autonomic multi-agent systems, particularly those having large numbers of agents, have revealed the need for design structures and formalisms to support the construction of properties that emerge at the system level. Such emergence, like self-* behaviour, relies typically on intricate inter-agent interactions. This paper shows how the top-down incremental approach of Formal Methods can be used satisfactorily in that situation, by considering a case study in which agents adapt and autonomously achieve a given configuration.

1 Introduction

Autonomic multi-agent systems achieve their important self-* properties (of configuration, management and repair, for instance) by exploiting carefully configured agent interactions. But engineering such interactions, so that the correct self-* properties emerge at the system level with a reasonable degree of trustworthiness, is not easy. New formalisms and design patterns appear necessary.

A first conceptual step has been taken by Zambonelli and Omicini [17] who, by distinguishing three levels of abstraction (the *macro* (or system) level, the *meso* (or agent-interaction) level, and the *micro* (or individual agent) level), have set the scene for a *reductionist* approach: one that reveals the intricacies level by level, in the manner familiar from Physical Science. This is one of the approaches taken by Formal Methods in the less ambitious case of conventional information systems. Is it also applicable to autonomic multi-agent systems?

Opinion is divided. In this paper we summarise the various arguments (in Section 2) then, in the body of the paper, consider a case study (Sections 3 to 6) chosen to exhibit typical self-* behaviour. Focusing on the property of self-reconfiguration, we choose an example from robotics in which mobile ‘atoms’ must assemble autonomously to achieve a certain global shape.

As an example of autonomous multi-agent systems the case study is particularly stringent: agents must conform to a given spatial target. The role of

* The first author acknowledges support from Australian Research Council (ARC) Discovery Grant DP0987452, *Combining Time Bands and Teleo-Reactive Programs for Advanced Dependable Real-Time Systems*.

** The second author acknowledges support from the Macao Science and Technology Development Fund under the PEARL project, grant number 041/2007/A3.

agents is played by the atoms, the micro-level behaviour is that of the atoms individually, the meso-level behaviour—of primary importance in this paper—consists of inter-atom interactions, and the emergent macro behaviour is the formation of the given shape. The techniques involved, however, seem relevant in achieving the weaker properties of ‘subgrouping’ or ‘clustering’ required by many meso-level interactions. Techniques for achieving this can be expected to play the dynamic equivalent, for multi-agent systems, of the *design patterns* so useful in object-oriented systems. The algorithm we treat is a modification from a family of algorithms given by Støy [12]. In the current context its adaptability (or ‘self-repair’ in Støy’s terminology) is important.

A feature of the incremental top-down approach taken here is that an operation considered to be atomic at one level may be split at the next level down. That idea is far from new. Our use of it has been partially motivated by the ‘time bands’ approach [7,2] in which a split is seen as resulting from temporal considerations. An important by-product of our approach is the clarification of conditions that ensure correctness of the algorithm (or ‘convergence’ in robotics terminology). The way in which they emerge is particularly satisfying.

2 Related Work

It has been suggested in the literature that the top-down approach of Formal Methods is not applicable to multi-agent systems. One argument is that the models produced are necessarily incomplete and therefore do not accurately reflect the real world¹. This argument, based on Gödel’s incompleteness theorem, applies equally to models for simulation as it does to mathematical models, and equally to single-component interactive systems as it does to multi-agent systems. Hence, it is not a new problem, and the usefulness of (possibly incomplete) models in these other areas indicates that they should also be useful for multi-agent systems. This point of view is supported by recent work on mathematical modelling of systems which has shown that such approaches are useful despite the necessary use of simplifying assumptions [16,6,3].

Another argument is that proofs over such models cannot be totally automated, nor can their subsequent development towards implementations be totally automated [5]. However, this does not preclude use of formal methods. It is well known that while automation can be used to support formal methods, high-level proof strategies and design decisions need to be developed by the engineer. Indeed, this is essential if we are fully to understand the design and proofs and hence have any real confidence in them.

Finally, a reluctance to adopt formal methods has arisen from the similarities between multi-agent systems and complex systems. Formal approaches have found less utility in the field of Complex Systems due to the emphasis on *observation* of existing systems (both natural and man-made, *e.g.*, the Internet)

¹ The argument is elaborated in [15] and referenced by several papers in the literature on multi-agent systems, *e.g.*, [17].

and the *prediction* of their global behaviour. When we model an existing system, unknown discontinuities in behaviour may not be modelled, and hence proof techniques may not be successful in uncovering emergent behaviour. Furthermore, it is claimed that some complex systems exhibit *strong emergence* [1] (e.g., the mind) and therefore, by definition, proofs cannot be constructed of how their behaviour arises.

With multi-agent systems, however, the emphasis changes from observation of existing systems to *engineering* of new systems and from prediction to *assurance* of their global behaviour. When we engineer new systems, we are not trying to prove the existence of emergent behaviours. Rather we start with the emergent behaviour we require (which may include the avoidance of undesirable behaviours), and develop a design which ensures it. To provide assurance that a design is sufficient, the emergent behaviour must be a consequence of the component interactions within the design. Hence, in this context we are interested only in systems which exhibit *weak emergence* [1]. This approach has been suggested and discussed in the context of the InterLink² programme [8,9].

There is little work on a general approach to formal assurance of multi-agent systems. The most mature work in the area focuses on particular systems [16,6,3]. Other papers suggest more general strategies, but do not apply them to significant case studies [10,18]. This paper is a first step in bridging this gap.

3 Case Study

In this section, we give an informal description of our case study: a collection of components referred to as ‘atoms’ which can configure to form 3-dimensional shapes through local interaction. The description is based on a family of solutions to this problem studied by Støy (and Nagpal) [13,14,12,11].

3.1 Atoms

Atoms in the system are capable of local communications with immediate neighbours, of movement over or around their neighbours, and of storing information including:

- a representation of the shape to be formed (referred to as the *target*);
- their position within the target, should they reach it;
- their status which is either *stationary* or *moving*;
- their distance from an atom in the target requiring a neighbour;
- their distance from a ‘seed’ atom used to start the shape-forming process.

Initially, only the seed atom stores the target and its position within it. It uses this information to decide at which of its neighbouring positions it needs atoms, and broadcasts accordingly. Atoms which respond to its broadcast for

² InterLink is an international coordination activity of ERCIM (European Research Consortium on Informatics and Mathematics).

neighbours are given the target and their position in it by the seed atom when they take up target positions. They then broadcast for the neighbours they require; and so on.

An atom has *stationary* status unless it is moving, or has decided to move, in response to a broadcast. Only stationary atoms store the distance from an atom requiring neighbours and from the seed atom. When an atom starts moving it will delete any such values it has stored. How these values are set up and utilised is described below.

3.2 Recruiting Neighbours

An atom moves towards the source of a request for neighbours by following a *recruitment gradient* established by the broadcaster. This gradient consists of an integer value at each atom, denoting the distance of the atom from the source of a broadcast.

An atom which broadcasts its need for neighbours sets its value of the recruitment gradient to 0 and sends this value to each of its neighbours. When an atom receives such a value, if it is either moving or decides to start moving in response to the receiving the value, it does nothing further with the value. Otherwise, if it does not have a value for the recruitment gradient, or has a value greater than that received, it sets its value to one more than the value received. It then sends this new value to all of its neighbours. If, on the other hand, it already has a recruitment gradient value less than that received, it ignores the received value.

This results in stationary atoms storing a value representing the shortest distance to an atom whose broadcast for neighbours has reached it. If it is able to follow a gradient to successively smaller values, a moving atom can reach the source of a broadcast and join the target shape; it then becomes stationary and remains so.

3.3 Connectedness

Initially, the atoms form a connected mass. This guarantees that a communication from one atom can reach any other atom in the system via a sequence of communications between neighbours. To ensure that the mass stays connected, each stationary atom stores an integer value denoting its distance from the seed atom. An atom is allowed to move only when it is certain that none of its neighbours rely on it to remain connected. That is, an atom can move only if its neighbours' distances from the seed do not exceed its own.

This *connection gradient* is set up by the seed atom before it begins broadcasting for neighbours. At this point in time, all atoms are stationary. The seed atom sets its own value to 0, and broadcasts this value to all of its neighbours. On receiving such a value, if an atom has no connection gradient value or a value greater than the received value, it sets its value to one more than the received value, and broadcasts this new value to its neighbours. If it has a value less than the received value, it ignores the received value.

3.4 Ensuring Progress

To ensure progress towards the final shape, we require that all broadcasts for neighbours reach available atoms. Connectedness alone does not ensure this as the recruitment gradient decreases only to the nearest source of a broadcast. We require also that, once an atom has all the neighbours it needs, its broadcast for neighbours is ‘cancelled’ (thus allowing other broadcasts to propagate). One way of doing that is to have all recruitment gradient values ‘dissipate’ after a certain time limit, and have broadcasters reestablish their individual gradients (after the same time limit) only if they still require neighbours.

Additionally, for progress we require that it is always possible for at least one atom not positioned in the target to reach the source of a broadcast. This is possible if the target has spaces through which atoms can move. Støy presents two algorithms to deal with that situation. In the first, a particular *scaffold* structure is used to tile (an approximation to) the target shape [12,11]. In the second, atoms of *varying resolution* are used, lowest first, to approximate the target [13]. Our treatment assumes instead that the target has sufficient space around it to allow the uninhibited movement of atoms.

Finally, we require that there are enough atoms to form the desired target shape. We assume that the number of atoms in the system are known by the user who chooses a target with exactly that number of atoms.

4 Macro-level Specification

In this section, we specify the case study at the macro-level. We choose the granularity of our initial level (or time band) so that the entire process of forming the desired target shape is accomplished in one event. The outcome of this event is that all atoms lie within the target. We then present two lower levels (*i.e.*, time bands with finer granularity) which progressively bring our specification to a form more suited for exploring a meso-level strategy.

4.1 Level 1

As a preliminary to our specification, we define a type *Position* denoting (absolute) positions in space. This type is not constrained in any way. We could be more precise and define position to consist, for example, of the set of all triples of real numbers. For our purposes, however, the more abstract definition suffices.

The (macro-level) system includes a *target* which is a finite set of positions (*i.e.*, of type $\mathbb{F} Position$) and a second finite set of positions, *atoms*, denoting the current position of the atoms. At this level we abstract the internal details of atoms, as well as any meso-level concepts such as gradients. For simplicity, we constrain the number of atoms to be equal to the number of positions in the target configuration (but return to consider an alternative in Section 6). We also have a variable, *placed*, to represent the set of atoms fixed in a location in the target.

$\textit{System1}$ <hr/> $\textit{target}, \textit{atoms}, \textit{placed} : \mathbb{F} \textit{Position}$ <hr/> $\#\textit{target} = \#\textit{atoms}$ $\textit{placed} \subseteq (\textit{target} \cap \textit{atoms})$

We then define the operation *Event1* to capture our desired goal: placement of an atom at each target position. (ΔS is a standard Z abbreviation to introduce the pre-state and post-state variables of an operation on state *S*, the latter being primed; thus the pre-state of the target is *target* and the post-state is *target'*.)

$\textit{Event1}$ <hr/> $\Delta \textit{System1}$ <hr/> $\textit{placed}' = \textit{target}'$ $\textit{target}' = \textit{target}$
--

At this level a single event occurs: *Event1*. In one step the atoms conform to the target (since the constraints imply that in fact $\textit{atoms}' = \textit{target}$).

4.2 Level 2

As a first step in our formal development, we introduce a lower level of granularity in which with each event some (but not necessarily all) of the atoms move into a position of the target. The final configuration is hence reached after several events. Once atoms are part of the target, they remain so.

We replace *Event1* in the previous specification with *Event2*. This operation specifies that the set of atoms fixed in the target before the operation (*placed*) forms a proper subset of those after the operation.

$\textit{Event2}$ <hr/> $\Delta \textit{System1}$ <hr/> $\textit{placed} \subset \textit{placed}'$ $\textit{target}' = \textit{target}$

One way to think of *Event2*, following the time-bands approach, is that the single time step of Level 1 is partitioned into finer time steps and *Event2* concerns the atoms that are placed within those finer steps. According to that view, for this to be a valid development step, we need to show that a finite repetition of *Event2*'s eventually achieves the post-state of *Event1*: for some $k : \mathbb{N}$,

$$\textit{Event1} = (\textit{Event2})^k. \quad (1)$$

That follows since either (i) the result has already been achieved and the iteration is vacuous, or (ii) since *Event2* strictly increases *placed* (and *target* is finite) within a finite number of iterations, *atoms* equals *target*. The formalisation of this proof (like later proofs in the paper) is straightforward and omitted.

4.3 Level 3

At the next level of granularity, each event corresponds to a single atom joining the target configuration.

We replace *Event2* with *Event3*. This operation specifies that a position in *target* but not in *placed* (i.e., in $target \setminus placed$) is added to *placed*.

$\frac{Event3}{\Delta System1}$
$\exists p : target \setminus placed \bullet placed' = placed \cup \{p\}$ $target' = target$

We need to show that a sequence of occurrences of *Event3* corresponds to a single *Event2* occurrence. This follows since *Event3* is an instantiation of *Event2* where the increase in the size of *placed* is 1. More precisely,

$$Event1 = (Event3)^k \tag{2}$$

where $k : \mathbb{N}$ is the difference in size between *target* and the (so far unspecified) initial value of *placed*.

It is possible in Z to adopt (from process algebra) an *interleaving* semantics where events that may occur in any order may be implemented concurrently. Such a semantics would be necessary for this specification to allow implementations where finite numbers of *Event3*'s occur in parallel.

5 Meso-level Specification

In this section, we extend the macro-level specification of Section 4 to include meso-level design. The final macro-level specification has focused our attention on a single atom moving to a position in the target. We introduce a lower level of granularity in which this event corresponds to two events: an atom within the target broadcasting its need for a neighbour, and an atom not within the target responding to the broadcast by moving towards the atom in the target.

At a lower level, we then introduce Støy's notions of gradients to realise the broadcast and moving events. It is at this level that the formal approach brings to the fore the necessary constraints on the system. To prove the development step is valid, it is necessary that the atoms form a connected structure, and that the target configuration allows atoms to move about within it.

5.1 Level 4

So far we have abstracted the mechanism by which atoms are placed.

At this level, we introduce local communication between atoms. The definition of 'local' depends on whether we are talking about programmable matter, where only neighbours in contact can communicate, or swarm systems, with wireless communication. For generality, we define a relation $in_range : Position \leftrightarrow$

Position such that an atom at position p can (directly) communicate with an atom at q if, and only if, $(p, q) \in in_range$. From now on, ‘neighbour’ means ‘neighbour with respect to in_range ’.

We introduce a new state variable *waiting* : $\mathbb{F} Position$ to consist of those placed atoms having communicated that they require one or more neighbours.

$\underline{\textit{System4}}$
$\textit{System1}$ $waiting : \mathbb{F} Position$
$waiting \subseteq placed$

For the algorithm to work, we need at least one placed atom initially. We assume there is exactly one such ‘seed’ atom. Initially, this atom has not broadcast its need for neighbours, and hence *waiting* is empty.

$\underline{\textit{Init4}}$
$\textit{System4}$
$\#placed = 1$ $waiting = \emptyset$

We replace *Event3* with *Move4* which strengthens the former operation by ensuring that the atoms move next to a ‘waiting’ atom in the target.

$\underline{\textit{Move4}}$
$\Delta\textit{System4}$
$\exists p : waiting; q : target \setminus placed \bullet$ $in_range(p, q) \wedge placed' = placed \cup \{q\}$ $target' = target \wedge waiting' = waiting$

That operation is enabled, however, only if an atom in the set *waiting* has a target neighbour not in *placed*. That fact is recorded in Z as the precondition of the operation *Move4* (fat parentheses denote relational image):

$\underline{\textit{pre Move4}}$
$\textit{System4}$
$in_range(\downarrow waiting) \cap (target \setminus placed) \neq \emptyset$

Hence, in order to ensure that *Move4* becomes enabled, we need an additional operation, *Broadcast4*, to add positions to *waiting*.

$\underline{\textit{Broadcast4}}$
$\Delta\textit{System4}$
$\exists p : placed; q : target \setminus placed \bullet$ $in_range(p, q) \wedge waiting' = waiting \cup \{p\}$ $target' = target \wedge atoms' = atoms \wedge placed' = placed$

That is in turn enabled whenever a placed atom has a target neighbour not in *placed*:

$$\text{in_range}(\text{placed}) \cap (\text{target} \setminus \text{placed}) \neq \emptyset. \quad (3)$$

Broadcast4 does not change the state variables of the previous specification, *atoms*, *target* and *placed*. Hence, it is equivalent to a ‘skip’ operation at that level. Therefore *Event3* is achieved by sufficiently many *Broadcast4* events followed by a consequently enabled *Move4*. Writing ‘ \circ ’ for sequential composition and * for its transitive closure, $\text{Event3} = (\text{Broadcast4})^* \circ (\text{Move4})$. So from (2)

$$\text{Event1} = ((\text{Broadcast4})^* \circ (\text{Move4}))^k \quad (4)$$

where $k = \# \text{target} - 1$ is the difference in size between *target* and the initial value of *placed*.

5.2 Level 5

At this level, we introduce recruitment gradients explicitly and model the mechanisms for propagating and following them. We add a state variable *grad* to denote the strength of the recruitment gradient at each atom to which it has propagated. Thus *grad* is a partial function (denoted by the symbol \mapsto) on *atoms*.

As the variable *waiting* is no longer required at this level, we extend the state *System1* rather than *System4*.

$\begin{array}{l} \text{System5} \\ \text{System1} \\ \text{grad} : \text{Position} \mapsto \mathbb{N} \\ \hline \text{dom grad} \subseteq \text{atoms} \end{array}$
--

Initially, no gradients have been propagated and so the gradient is undefined at each atom.

$\begin{array}{l} \text{Init5} \\ \text{System5} \\ \hline \# \text{placed} = 1 \\ \text{grad} = \emptyset \end{array}$

If a placed atom at position *p* requires one or more neighbours, it creates a new gradient having strength 0 at *p*. The precondition for being able to do so is simply (3). That strength replaces any previous gradient strength at *p* (described below using Z’s functional overriding operator \oplus).

$\begin{array}{l} \text{CreateGrad5} \\ \Delta \text{System5} \\ \hline \exists p : \text{placed}; q : \text{target} \setminus \text{placed} \bullet \\ \text{in_range}(p, q) \wedge \text{grad}' = \text{grad} \oplus \{p \mapsto 0\} \\ \text{target}' = \text{target} \wedge \text{atoms}' = \text{atoms} \wedge \text{placed}' = \text{placed} \end{array}$
--

An atom at position p can propagate a gradient to a neighbouring atom at q provided q has no gradient (*i.e.*, is not in the domain of $grad$) or has a gradient strength less than the gradient strength at p plus one.

$\frac{PropGrad5}{\Delta System5}$
$\begin{aligned} &\exists p : placed; q : atoms \bullet \\ &\quad in_range(p, q) \wedge \\ &\quad (q \notin \text{dom } grad \vee grad(q) < grad(p) + 1) \wedge \\ &\quad grad' = grad \oplus \{q \mapsto grad(p) + 1\} \\ &target' = target \wedge atoms' = atoms \wedge placed' = placed \end{aligned}$

An atom at $p \notin placed$ can follow a decreasing gradient towards its origin (with the aim of becoming a neighbour of the atom which created the gradient). In the following description the atom at p is initially a neighbour of that at q , whose position afterwards, p' , makes it a neighbour of the atom at r which is a neighbour of that at q but having lower gradient strength.

$\frac{FollowGrad5}{\Delta System5}$
$\begin{aligned} &\exists p : atoms \setminus placed; q, r : \text{dom } grad \bullet \\ &\quad in_range(p, q) \wedge in_range(q, r) \wedge grad(r) < grad(q) \wedge \\ &\exists p' : Position \setminus atoms \bullet in_range(p', r) \wedge atoms' = (atoms \setminus \{p\}) \cup \{p'\} \\ &target' = target \wedge placed' = placed \wedge grad' = grad \end{aligned}$

Since this operation does not change $placed$, another operation is required to fix atoms in the target. An atom not in $placed$ can join as a neighbour of an atom with gradient strength zero. If the latter atom does not require further neighbours, it is removed from the gradient (\Leftarrow is Z's notation for domain subtraction).

$\frac{JoinTarget5}{\Delta System5}$
$\begin{aligned} &\exists p : atoms \setminus placed; q : \text{dom } grad \bullet \\ &\quad in_range(p, q) \wedge grad(q) = 0 \wedge \\ &\quad (\exists p' : target \setminus placed \bullet \\ &\quad \quad in_range(q, p') \wedge \\ &\quad \quad atoms' = (atoms \setminus \{p\}) \cup \{p'\} \wedge \\ &\quad \quad placed' = placed \cup \{p'\}) \wedge \\ &\quad (\nexists r : target \setminus placed' \bullet in_range(q, r)) \Rightarrow grad' = \{q\} \Leftarrow grad \wedge \\ &\quad (\exists r : target \setminus placed' \bullet in_range(q, r)) \Rightarrow grad' = grad \\ &target' = target \end{aligned}$

The invariant $\#target = \#atoms$ and the fact that $target$ does not change ensure that either $p' = p$ or p' is not already occupied.

When the gradient to a particular atom is no longer required (since the atom has all of its neighbours), the gradient must be removed from the system. This is necessary to enable other gradients to be followed; yet operation *JoinTarget5* removes only a source. Another operation is required to allow non-zero gradients to dissipate over time whenever they are not reinforced by *PropGrad5* events.

$$\frac{\text{Dissipate5} \quad \Delta\text{System5}}{\exists p : \text{dom } \textit{grad} \bullet \textit{grad}(p) \neq 0 \wedge \textit{grad}' = \{p\} \triangleleft \textit{grad} \\ \textit{target}' = \textit{target} \wedge \textit{atoms}' = \textit{atoms} \wedge \textit{placed}' = \textit{placed}}$$

Verification of this development step requires a data refinement [4] relating the variable *grad* to the variable *waiting* of the previous specification: a position *p* in which $\textit{grad}(p) = 0$ is related to a position in *waiting* which still requires neighbours.

$$\frac{\text{Rel} \quad \text{System4} \quad \text{System5}}{\forall p : \text{Position} \bullet \\ p \in \text{dom } \textit{grad} \wedge \textit{grad}(p) = 0 \\ \Leftrightarrow \\ p \in \textit{waiting} \wedge (\exists q : \textit{target} \setminus \textit{placed} \bullet \textit{in_range}(p, q))}$$

Subject to *Rel*, *CreateGrad5* performs the same task as *Broadcast4*:

$$\text{Broadcast4} = (\text{Rel} \wedge \text{CreateGrad5} \wedge \text{Rel}') \setminus (\textit{grad}, \textit{grad}') \quad (5)$$

where *Rel'* is *Rel* with all declared variables primed, $S \wedge T$ is the schema formed from schemas *S* and *T* by merging their declarations and conjoining their predicates, and $S \setminus (x, y)$ is the schema *S* with the variables *x* and *y* hidden.

Similarly, *JoinTarget5* performs the same task as *Move4*. To ensure that *JoinTarget5* is enabled whenever *Move4* is, however, we need to show that the other operations, *PropGrad5*, *FollowGrad5* and *Dissipate5* (which correspond to ‘skip’ operations at the previous level) lead to *JoinTarget5* being enabled; the intention is to replace the right-hand side of (4) by

$$(\text{Rel} \wedge (\text{Init5} \text{ ; } \text{CreateGrad5} \text{ ; } \mathcal{O})^k \wedge \text{Rel}') \setminus (\textit{grad}, \textit{grad}') \quad (6)$$

where \mathcal{O} is a parallel composition of a single *JoinTarget5* with a finite number of iterations of *CreateGrad5*, *PropGrad5*, *FollowGrad5* and *Dissipate5* operations. This will be true if (i) gradients propagate to atoms that are not yet in the target, and (ii) atoms can follow these gradients to their origin.

Attempts to prove (i) will immediately run into a problem. Propagation relies on atoms being neighbours and there is no guarantee that initially the set *atoms*

is connected. Furthermore, there is no guarantee that a connected collection of atoms remains connected after an atom changes position. To ensure it, we could add an invariant to *System5* stating that between any two atoms there lies a sequence of atoms, consecutive pairs of which are neighbours. Expressed in terms of transitive closure, the extra property of *System5* is:

$$connected \equiv ((in_range)^* = atoms \times atoms). \quad (7)$$

While this suffices to prove (i) (taking into account that gradients will eventually dissipate and allow other gradients to propagate when (ii) holds) it does not reflect an implementation in terms of local interactions. Although we could ensure the invariant holds initially, we need a local mechanism to ensure that it continues to hold when atoms move. Støy's solution to this problem is to add another, *connection*, gradient. An atom is allowed to move only when the connection gradient strengths of its neighbours do not exceed its own. If that were not the case, neighbours having higher gradient strengths might rely on the atom for connection to the rest.

We could specify such a connection gradient and its propagation just as we have specified the recruitment gradient. It would be created by the initial 'seed' atom, but would not dissipate. The formal details are routine and are omitted.

Attempts to prove (ii) will also lead to a problem. It is not guaranteed that an atom can follow a recruitment gradient through the existing target where the atoms are unable to move. Støy solves this problem in [12,11] by ensuring that a target is in the form of a 'scaffold' with spaces around the atoms already in fixed positions.

We could capture this formally by including an initial condition on *target*. Since *target* is unchanged by any operation, this condition is invariant:

$$lacunose \equiv \forall p : target \bullet \exists q : Position \setminus target \bullet in_range(p, q). \quad (8)$$

Such a condition ensures that the algorithm succeeds in filling the target (in Støy's notation, is *convergent*).

That completes the development at the meso level. The development of the micro level is far more detailed since atoms must contain sufficient state to keep track of their position, the target, their neighbours, the gradients, avoiding collisions, and so on. However it does not add to our understanding of atomic interactions, the focus of the current paper, and so is not considered here.

6 Adaptivity

In this section we evaluate the suitability of our approach for expressing adaptability, a feature that is even more important in the multi-agent system context than in Støy's original setting of reconfigurable robots.

An adaptive algorithm must respond to dynamic changes in its environment. We consider here the removal of target positions: at run time some target positions are blocked by the environment. If such a position already contains an

atom, the atom is removed from the system. Atoms can sense when a neighbouring position is blocked and are unable to move into that position.

The new system, *System5a*, differs from *System5* by satisfying the weaker invariant $\#target \leq \#atoms$ (since an atom is not always removed when a target position is blocked).

$\textit{System5a}$ <hr/> $target, atoms, placed : \mathbb{F} \textit{Position}$ $grad : \textit{Position} \leftrightarrow \mathbb{N}$ <hr/> $\#target \leq \#atoms$ $placed \subseteq (target \cap atoms)$ $\text{dom } grad \subseteq atoms$ $connected$ $lacunose$

Nonetheless, the condition $placed' = target'$ of *Event1* correctly describes what the system should now achieve, although its other condition, $target' = target$, must be weakened to an inclusion. The result fills the target with atoms but may leave some, connected to the others, outside the target.

$\textit{Event1a}$ <hr/> $\Delta \textit{System1}$ <hr/> $placed' = target'$ $target' \subseteq target$

Adaptation is modelled by a revision of (6), operating instead on *System5a* and including an operation *Block* (in parallel with the other operations of \mathcal{O} but the only one to change the target) which removes a set *blocked* of target positions and updates the system components accordingly. Explicitly:

\textit{Block} <hr/> $\Delta \textit{System5a}$ $blocked : \mathbb{F} \textit{Position}$ <hr/> $blocked \subseteq target$ $target' = target \setminus blocked$ $atoms' = atoms \setminus blocked$ $placed' = placed \setminus blocked$ $grad' = blocked \triangleleft grad$

It is enabled iff $blocked \subseteq target$ and $atoms \setminus blocked$ remains connected (since (7) is invariant). Since it only removes target positions, (8) still holds.

After *Block* occurs the recruitment gradient may be temporarily disrupted (since some sources may have been removed and other placed atoms may find they no longer need certain neighbours). However in spite of the inconsistency

between the representation of the target generated by the seed and that which now pertains, the recruitment gradient regenerates and succeeds in filling the new target, since the physical conditions *connected* and *lacunose* are preserved. The same holds even with multiple invocations of *Block*. Adaptability is captured by the claim that *Event1a* is achieved by the revised version of (6).

Adaptability to an increased target or decreased atom set is handled similarly. If a situation is too severe for Støy's algorithm (*e.g.*, an increased target violating (8)), the deficiency is revealed; if an extension is possible, *placed'* emerges.

7 Conclusion

We have presented a case study to support the view that formal methods can be used in a top-down incremental manner to account for behaviour which emerges at the system level of a multi-agent system. One strength of the approach has been the ease with which the physical conditions of connectedness and lacunosity arise as necessary for correctness. Another has been the manner in which the meso-level design structures (the recruitment and connectivity gradients) have been revealed incrementally to facilitate inter-agent interactions and hence to achieve the required emergent behaviour.

Our case study indicates that emergence needs to be dealt with primarily at the boundary between the macro and meso levels. Conformance between specifications at these levels amounts to showing that a particular interaction paradigm gives rise to a desired global behaviour.

A further, crucial, benefit has been the ease with which the approach is able to incorporate adaptability. Whilst it makes no sense to discuss adaptability at our highest level of abstraction with its single, atomic operation, it becomes pertinent at the meso level where the environment may supervene with its own atomic operation. For the system to be entirely adaptable that operation should have no precondition; in our case, it must maintain connectedness and lacunosity.

We believe that adaptability needs to be dealt with primarily at the boundary between the micro and meso levels. Adaptability is modelled here as response to an environmentally-induced operation changing state. So, of interest is how meso-level strategies for obtaining global behaviour under such disturbances arise from changes in agent behaviour in response to those conditions.

To assist engineers in the development of multi-agent systems using our approach, we would ideally have a catalogue of "patterns", similar to those used in object-oriented design, to facilitate the modelling process at each level. Furthermore, we would have a catalogue of "strategies" for proving conformance between specifications at different levels of abstraction. Identifying and formalising such patterns and strategies is an area of future work.

References

1. Bedau, M.A.: Weak emergence. In: Tomberlain, J. (ed.) *Philosophical Perspectives: Mind, Causation, and World*, vol. 11, pp. 375–399. Blackwell Publishers, Malden (1997)

2. Burns, A., Hayes, I.J., Baxter, G., Fidge, C.J.: Modelling temporal behaviour in complex socio-technical systems. Technical Report YCS 390, University of York (2005)
3. Cucker, F., Smale, S.: On the mathematics of emergence. *Japanese Journal of Mathematics* 2, 197–227 (2007)
4. Derrick, J., Boiten, E.: *Refinement in Z and Object-Z: Foundations and Advanced Applications*. Springer, Heidelberg (2001)
5. Edmonds, B., Bryson, J.: The insufficiency of formal design methods — the necessity of an experimental approach for the understanding and control of complex MAS. In: *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, pp. 938–945. IEEE Computer Society, Los Alamitos (2004)
6. Hamann, H., Wörn, H.: A framework of space-time continuous models for algorithm design in swarm robotics. *Swarm Intelligence* 2, 209–239 (2008)
7. Henzinger, T.A., Qadeer, S., Rajamani, S.K.: Assume-guarantee refinement between different time scales. In: Halbwegs, N., Peled, D.A. (eds.) *CAV 1999*. LNCS, vol. 1633, pp. 208–221. Springer, Heidelberg (1999)
8. Jun, H., Liu, Z., Reed, G.M., Sanders, J.W.: Ensemble engineering and emergence. In: Wirsing, M., Banâtre, J.-P., Hölzl, M., Rauschmayer, A. (eds.) *Challenges for Software-Intensive Systems and New Computing Paradigms*. LNCS, vol. 5380, pp. 162–178. Springer, Heidelberg (2008)
9. Sanders, J.W., Smith, G.: Formal ensemble engineering. In: Wirsing, M., Banâtre, J.-P., Hölzl, M., Rauschmayer, A. (eds.) *Challenges for Software-Intensive Systems and New Computing Paradigms*. LNCS, vol. 5380, pp. 132–138. Springer, Heidelberg (2008)
10. Stepney, S., Polack, F., Turner, H.: Engineering emergence. In: *IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2006)*, pp. 89–97. IEEE Computer Society, Los Alamitos (2006)
11. Støy, K.: Controlling self-configuration using cellular automata and gradients. In: *8th International Conference on Intelligent Autonomous Systems (IAS-8)* (2006)
12. Støy, K.: Using cellular automata and gradients to control self-reconfiguration. *Robotics and Autonomous Systems* 54, 135–141 (2006)
13. Støy, K., Nagpal, R.: Self-reconfiguration using directed growth. In: *7th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, pp. 149–160. ACM Press, New York (2004)
14. Støy, K., Nagpal, R.: Self-repair through scale independent self-reconfiguration. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2062–2067. IEEE Press, Los Alamitos (2004)
15. Wegner, P.: Why interaction is more powerful than algorithms. *Communications of the ACM* 40(5), 80–91 (1997)
16. Winfield, A., Liu, W., Nembrini, J., Martinoli, A.: Modelling a wireless connected swarm of mobile robots. *Swarm Intelligence* 2, 241–266 (2008)
17. Zambonelli, F., Omicini, A.: Challenges and research directions in agent-oriented software engineering. *Autonomous Agents and Multi-Agent Systems* 9(3), 253–283 (2004)
18. Zhu, H.: Formal reasoning about emergent behaviour in MAS. In: *International Conference on Software Engineering and Knowledge Engineering (SEKE 2005)*. Knowledge Systems Institute (2005)

Using Reinforcement Learning for Multi-policy Optimization in Decentralized Autonomous Systems – An Experimental Evaluation

Ivana Dusparic and Vinny Cahill

Lero–The Irish Software Engineering Research Centre
Distributed Systems Group
School of Computer Science and Statistics
Trinity College Dublin, Ireland
{ivana.dusparic,vinny.cahill}@cs.tcd.ie

Abstract. Large-scale autonomous systems are required to self-optimize with respect to high-level policies, that can differ in terms of their priority, as well as their spatial and temporal scope. Decentralized multi-agent systems represent one approach to implementing the required self-optimization capabilities. However, the presence of multiple heterogeneous policies leads to heterogeneity of the agents that implement them. In this paper we evaluate the use of Reinforcement Learning techniques to support the self-optimization of heterogeneous agents towards multiple policies in decentralized systems. We evaluate these techniques in an Urban Traffic Control simulation and compare two approaches to supporting multiple policies. Our results suggest that approaches based on W-learning, which learn separately for each policy and then select between nominated actions based on current action importance, perform better than combining policies into a single learning process over a single state space. The results also indicate that explicitly supporting multiple policies simultaneously can improve waiting times over policies dedicated to optimizing for a single vehicle type.

1 Autonomous Systems

Autonomic computing systems are systems that self-manage and self-adapt to varying circumstances without human intervention [8]. The need for autonomic capabilities arises due to the increasingly large scale, decentralization and complexity of computing systems rendering the traditional manual, centralized and hierarchical approaches to system management infeasible [21]. Autonomic systems are only given high-level objectives while the details of how to meet those objectives are left up to the systems themselves. Therefore, autonomic systems need to self-optimize their performance, even in the changing environment conditions. Rather than being managed by a central component, an autonomic system can be modelled as a group of autonomic elements, that are capable of sensing their environment and making their own local decisions [8]. The optimal local decisions cannot be predefined for each situation in which an autonomic element

might happen to be, but is often required to be learnt by the element itself. As autonomous agents [17] have the capabilities required by autonomic elements, it is believed that multi-agent systems are a suitable technique for the implementation of autonomic behaviour [21]. Examples of such techniques already successfully applied in decentralized large-scale systems include ant colony optimization in load balancing [10], particle swarm optimization in wireless networks [7], evolutionary computing in routing [5] and reinforcement learning (RL) in load balancing [4].

1.1 Multi-policy Optimization

The systems mentioned above focus on optimizing system performance with respect to only a single high-level goal. However, autonomic systems might often be required to meet multiple goals simultaneously. These goals can be expressed as system policies, which are used to guide system behaviour. Therefore, optimization techniques need to be able to address multiple goals (or policies) simultaneously. We hypothesize RL might be a suitable basis for the implementation of such a technique, as it has already been successful as a learning technique for optimization towards a single policy in decentralized systems, as well as a learning technique for multiple policies on a single agent (see Section 2). We test our hypothesis in a simulation of an Urban Traffic Control (UTC) system. We believe UTC systems are representative of large-scale autonomic systems, as existing centralized techniques are failing to deal with the pressure of high traffic loads and new decentralized adaptive learning techniques are being investigated to deal with increasing traffic congestion (see Section 2). UTC systems may also need to optimize for multiple policies that have different characteristics. The policies can often be conflicting, highly dependent on one another, have different levels of priority, and different spatial and temporal scope.

Policy classification. We use three main criteria to classify the characteristics of policies in decentralized systems:

- priority - can range from low to high, based on how important it is for a system to meet this particular goal in relation to meeting its other goals;
- spatial scope - can be local, regional, and global, based on the area of a system over which a policy is implemented and its performance measured;
- temporal scope - can be continuous or temporary (sporadic), based on whether a system is required to work towards this goal continuously during its operation, or only occasionally under a certain set of conditions.

We illustrate the classification with a few examples of policies from UTC. The main task of a UTC system is to optimize global traffic flow in the system, by minimizing travel and waiting times for all vehicles in the system. This policy is classified as global (as it affects the whole system), continuous (as UTC systems need to implement this policy as long as there are any vehicles present in the system), and of a standard priority. Occasionally, emergency vehicles, such as ambulances, fire engines, or police cars, appear in the UTC system, and the

system's task is to give them priority over other vehicles in the system. This policy that prioritizes emergency vehicles is said to be regional (as it affects only the region in which the emergency vehicle is travelling, generally major traffic routes), sporadic (as emergency vehicles are not always present in the system but only when the need arises), and it has a high priority (since it is more important to meet this policy than to minimize the travel time for other vehicles on non time-critical journeys). Policies can also be local, where, for example, at a very busy pedestrian crossing, pedestrians may be given priority over vehicles.

Agent heterogeneity and dependency. This wide variety of policies and their characteristics leads immediately to heterogeneity of the agents that implement them. For example, consider Figure 1. Agent A might be in charge of contributing to the implementation of a global policy P_a , together with all of the other agents in the system. Agent B could also be in charge of contributing to the implementation of a policy P_b , implemented only by agent B and its neighbours, while agent C could also be in charge of a local policy P_c , being the only agent implementing it. These policies can be concerned with addressing different parts of the environment, e.g., P_a might only be interested in optimizing travel time for cars, while P_c might only be dealing with pedestrians. In terms of RL, this will cause the state spaces of agents A, B, and C to differ, as, for example, the information required to be encoded in the state space of a policy optimizing for cars will need to be different from the information relevant to the policy optimizing for pedestrians.

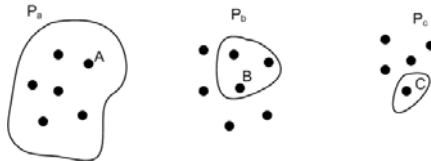


Fig. 1. Agent heterogeneity

Furthermore, agent heterogeneity can also arise from the differences in the agents' environments and capabilities. For example, in a UTC system, the layout of junctions can differ; each junction can have a different number of approaches and exits, resulting in a different set of traffic-light phases being possible at that junction. In an RL implementation this maps to agents having different state spaces as well as different action sets. A junction with two approaches and one exit will have a significantly smaller state space and action set than a junction with four approaches and four exits. The size of the state space and the number of possible actions directly influence the duration of a learning process, so agents will significantly differ in the number of learning steps that it takes them to learn what they consider to be the optimal action for each state visited.

Agents acting in a shared environment may potentially be highly dependent, i.e. affected by each other's actions. In UTC, agents share the same road network

with limited road space. Therefore, any decision that a traffic-light agent makes might have direct consequences on its neighbouring agents, and by extension, on most of the other agents in the system. For example, if a backlog of traffic is left uncleared at an approach, the queue can spill over to the upstream junction. The traffic will not be able to go through the upstream junction regardless of the actions taken by an agent controlling it, as there is no road space available.

Such dependencies are particularly difficult to deal with in environment controlled by heterogeneous agents. Some agents could be contributing towards optimizing traffic flow, while others are contributing to prioritizing emergency vehicles. However, cars and emergency vehicles share the same road network, and therefore the performance of agents in implementing one policy will directly influence the performance towards the other. For example, if an agent that is implementing emergency vehicle prioritization releases an approach at which an emergency vehicle is waiting, it might create a traffic backlog on one of its other approaches, negatively affecting the junction upstream from that approach.

In summary, large-scale autonomic systems can consist of multiple agents, implementing multiple, dependant, and potentially conflicting policies, where these policies differ between agents, causing agents to have different state spaces and different action sets. All of these issues will need to be addressed by RL techniques that are to be applied to large-scale autonomic systems and UTC in particular.

1.2 Research Question

The goal of this study is to assess the suitability of multi-agent RL-based techniques for optimization in autonomic systems. In order to do so, we have implemented and evaluated several single and multi-policy UTC scenarios. We use single-policy scenarios to evaluate the impact that policies targeted at one vehicle type have on other vehicle types, as well as baselines for the evaluation of multi-policy scenarios.

As policy heterogeneity is a central issue, for the initial evaluation we selected two policies that differ in all three of our classification criteria: priority, temporal scope, and spatial scope. The single-policy scenarios we implemented are as follows:

1. Global Waiting time Only (GWO) - a global, continuous, standard-priority policy that aims to optimize waiting time for all the vehicles in the system.
2. Emergency Vehicles Only (EVO) optimization - a regional, temporary, high-priority policy that aims to prioritize emergency vehicles only.

We combined the policies above in two ways to implement the following multi-policy scenarios:

1. Combined state space (GWEV-c), where GWO and EVO are combined into a single learning process over a single state space.
2. W-Learning (GWEV-w), where GWO and EVO learn the best actions separately as two separate learning processes, but W-learning (see Section [2.1](#)) is used to determine which action is to be executed.

One important consideration when addressing the agent dependency and heterogeneity in large-scale autonomic systems is whether agents should act independently (contributing only to implementing policies for which they are responsible), or whether they should collaborate with other agents (contributing to the implementation of policies that they are not directly responsible for as well). In the experiments we describe in this paper, we implement only independent agents. The scenarios above are designed to evaluate approaches for dealing with multiple heterogeneous policies, while in the future, we also plan to evaluate the impact of collaboration in multi-agent multi-policy approaches.

The rest of this paper is organized as follows. In Section 2 we give background on RL as well as its applications in UTC system. Section 3 describes our simulation environment. Section 4 describes the details of the scenarios that we implemented and the design of the agents, followed by the results and their analysis. Section 5 concludes the paper and outlines future work.

2 Background

Reinforcement Learning [20] is an unsupervised learning technique whereby an agent learns how to meet its goal by interacting with the environment. Agents sense their environment, map their observations to a state space representation, execute an action and obtain a reward from the environment based on the suitability of that action in the given state. Therefore, a reward is the only guidance agents have when learning how to meet their objectives. We are particularly interested in Q-learning implementations of RL [20], because, as we'll see later in this section, it has already been successfully applied to certain types of UTC problems. In Q-learning, an agent uses a value function to estimate the accumulated future reward. In this way, agents learn to perform the actions with the highest long-term reward, rather than those that merely receive the highest immediate reward. Performance of a Q-learning process depends on the action selection strategy used. In our experiments we use Boltzmann [20] action selection, which uses a temperature parameter to determine the ratio of exploration and exploitation in the Q-learning process. The speed of learning and the weight given to recent vs. older actions are determined by two additional parameters, the learning rate α , and a discount factor γ , respectively.

2.1 Multi-goal Q-Learning

Q-learning implementations can deal with the presence of multiple policies on a single agent in several ways.

Humphrys [6] introduced W-learning, where policies not only learn appropriate actions for each state, but also how important it is to that policy for that particular action to be executed, in comparison to an action that is best for some other policy. The action with the highest relative importance gets executed.

In contrast to W-learning, multiple goals can also be combined into a single learning process. However, on a larger scale this is prone to state explosion.

One way to deal with this is to reduce the state space by eliminating states that are unlikely or impossible to occur, if there are any [3].

Nataraj et al. [11] deal with learning in the situations where relative weights of policies change over time. Shelton provides a means to balance the incomparable rewards received from multiple sources [19].

All of these multi-policy techniques have only been applied to a single agent.

2.2 Agent-Based Traffic Control Strategies

A large body of research exists showing the suitability of multi-agent systems, in particular those implementing RL, for optimization of performance of UTC systems. An increasing number of UTC systems are being managed by traffic-responsive algorithms, such as SCATS [13], however research shows that use of RL can outperform these algorithms as well.

In [22], cars estimate their projected waiting time, and make a decision about their route based on this. Projected waiting time is communicated to the traffic lights agents, modelled as Q-Learning agents, that then select the phase that minimizes waiting times. Wiering's experiments show improved performance of his approach over a fixed-time controller. Abdulhai et al. [1] model traffic-lights as Q-learning agents and conclude that Q-learning provides higher real-time adaptivity than other state-of-the-art techniques. Pendrith [14] models vehicles as Q-learning agents capable of sensing the speed and position of their neighbouring vehicles. Based on this information, vehicles decide on their speed and potential lane changes.

In the above examples, RL agents are implemented to act independently. Bazan introduces traffic-light agent coordination using evolutionary game theory [2], while Salkham et al. [18] implement agent collaboration by a means of Q-value exchange between neighbouring agents.

All of the implementations mentioned are concerned with a single policy of optimizing traffic in general, whether by increasing throughput or by minimizing waiting time. Much less work is dedicated to the applicability of multi-agent systems, and RL in particular, for optimization towards other UTC policies. Oliveira and Duarte [12] incorporate emergency vehicle priority into their UTC system. Meignan et al. [9] simulate bus network performance, modelling both buses and passenger behaviour. They account for the influence of other traffic on the bus network, but do not account for the influence of public transport on other vehicles, nor do they include traffic signal priority for public transport.

2.3 Summary

RL is a single-agent, single-policy learning technique. It has been extended to deal with multiple policies on a single agent, as well as to multiple agents implementing a single policy, either independently or collaboratively. However, no RL technique deals with multiple policies on multiple agents simultaneously. Existing RL applications in UTC also concentrate on a single policy only. Our work has been inspired by the success of such applications in optimization of

traffic waiting times, emergency vehicle priority, and bus networks performance individually, as well as by the lack of an integrated approach for optimizing for all policies and vehicle types simultaneously. We evaluate the use of RL techniques in multi-agent environments in dealing with multiple UTC policies, while simultaneously modelling the influence that these policies have on one another’s performance.

3 UTC Simulation Platform

In our experiments we use an urban traffic simulator developed in Trinity College Dublin [15]. The simulator uses a microscopic traffic simulation approach, and can simulate traffic over any road network defined by a map provided in an XML format. The simulator can distinguish between multiple vehicle types, such as cars, public transport vehicles, and emergency vehicles. Vehicles implement different behaviours based on their type; e.g. emergency vehicles are capable of driving above the allowed speed limit, as well as driving through red lights if it is safe to do so.

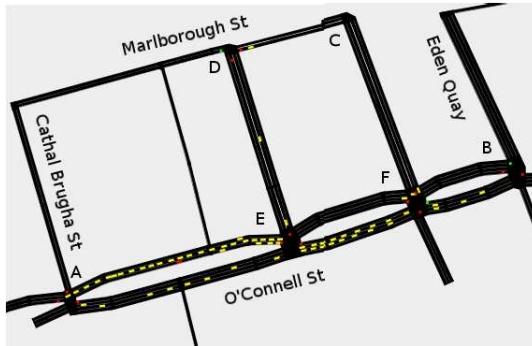


Fig. 2. UTC Simulator

The map we used for our initial experiments presented in this paper is shown in Figure 2. The map is based on road layout details provided by Dublin City Council and corresponds to one of the busiest areas of Dublin’s road network, O’Connell Street, Dublin’s main street, and several side roads that feed traffic onto this road. Using a real-world map provides a more realistic simulation; many of the simulations used for evaluation of multi-agent systems in UTC covered in Section 2.2 use either a single junction, or multiple junctions and road links that have similar layouts, while the map that we use includes junctions of various layouts (e.g. junctions with two, three, and four approaches and exits), roads of differing width (e.g. two, three, and four lane roads), as well as one-way and two-way roads. The map covers 8 junctions, 5 of which are signalised junctions and are controlled by the agents described in the following section. Each agent has a set of available phases, or combinations of compatible red and green settings

on all traffic lights controlling one intersection. Phases are generated based on intersection layout and allowed traffic directions. Each phase is mapped to an action agent is able to execute.

4 Traffic Light Agents

In this section we describe the agents that we designed to implement the policies and study the RL approaches to be evaluated (as listed in Section 4.2), as well as agents used as a basis for comparison.

4.1 Baseline Agents

Round robin. As a baseline used for the evaluation of the performance of the RL agents we ran experiments using a Round Robin (RR) junction controller. The RR agent, at each junction, continuously loops through all available phases at that junction. The duration of each phase is 20 seconds.

SAT. We also compare the performance of the RL agents to a simple SCATS-like traffic-responsive algorithm SAT, as defined by Richter [16], that adjusts phase duration based on the degree of saturation at a junction. The degree of saturation is defined as a ratio of the effectively used green time to the total available green time. At each junction, SAT, similarly to SCATS, aims to keep the junction saturation as close to 90% as possible, by shortening or lengthening the phase duration. In our experiments the minimum duration for each phase is set to 20 seconds.

4.2 Single-Policy RL Scenarios

GWO - Optimizing global waiting time. The first policy we implemented, optimizing Global Waiting time Only (GWO), optimizes vehicle waiting time in the whole system. Since global waiting time is a sum of waiting times for all cars at all junctions in the system, and we assume no collaboration between agents, we aim to minimize the waiting times at each individual junction.

Each agent is capable of sensing the number of vehicles at each of its approaches, and maps that to a state space that orders approaches according to their congestion. For example, on a junction with two approaches, a_1 and a_2 , a state could be “Congestion order: a_1, a_2 ”, meaning that approach a_1 has more traffic waiting than a_2 . Note that, since junctions have different layouts, the size of the state space will depend on the number of approaches. The state space does not encode how many vehicles are waiting at which approach as the numbers are relative to overall congestion in the system. It also contains information about whether the total number of vehicles waiting at the junction is more or less than at the previous phase change (e.g. “Congestion order: a_1, a_2 , less vehicles than before”). Note that arrival rates are assumed to be uniform in this experiment, so a change in numbers of vehicle waiting is caused only by an agent’s action. Such a state space is created to facilitate rewarding an agent (100 points in our

experiments) for being in a state with less traffic waiting than at the previous decision point, i.e. to motivate it to execute actions that clear more traffic than arrives at the junction during the action execution. Agents learn to reduce the number of vehicles waiting at the junction’s approaches, thus reducing global waiting time for the system.

EVO - Prioritizing emergency vehicles. The other single RL policy that we implemented minimizes waiting times for Emergency Vehicles Only (EVO). The agents’ state space encodes information about which approach(es), if any, have emergency vehicles waiting. Agents receive a reward (200 points in our experiments) for being in a state where there is no emergency vehicle present at any of the approaches. This encourages agents to, as soon as possible, return to the state with no emergency vehicle present, by enabling emergency vehicles to pass. This policy does not address any other vehicle types and only takes emergency vehicles into account when making action decisions.

4.3 Multi-policy RL Scenarios

GWEV-c: Merging RL processes. One way to combine multiple policies on a single agent is to encode all the information relevant for all the policies into a single state space and a single learning process. We combined GWO and EVO into a single policy, GWEV-combined (GWEV-c). The state space of GWEV-c consists of the cross product of the state spaces for GWO and EVO. An agent receives 100 points reward for being in any of the states with less traffic than in the previous phases (i.e. states for which GWO receives a reward for), a 200 points reward for any of the states with no emergency vehicles present (i.e. states for which EVO receives a reward for), and the sum of both rewards for being in a state that satisfies both criteria. We acknowledge that as the number of policies to be combined increases this approach will not be scalable due to state space explosion, but we believe that comparing its performance to other techniques can give us a useful insight into how multiple policies should be dealt with.

GWEV-w: W-learning. W-Learning is a multi-goal technique proposed in [6] that builds on Q-Learning. First, each agent runs separate Q-Learning process for each policy that it is implementing. In our experiments, we ran the two individual single goal policies described in the previous section, GWO and EVO, and on top of them implement W-learning (GWEV-w). After GWO and EVO have learnt Q-values for their state-action pairs, the process of W-learning starts. In W-learning, an agent learns how important it is that, for each of its policies and for each state in which an agent could be, the action a policy nominates is in fact executed, i.e. what weight that action carries. W-values are updated based on the reward received, and further action selection is based on these W-values. Each policy nominates an action, based on its Q-values, together with an associated W-value for the state in which the agent is currently. The action proposed by a policy with the highest W-value is executed. In our experiments, since EVO is a temporary policy, we deem it inactive when there are no emergency vehicles

present, and set the weight of the action that the EVO policy nominates in that state to zero.

4.4 Simulation Setup

Cars enter the simulation at four different points (A, B, C, D) and exit the system at two different points (A, B), following 1 of 4 paths: A to B, B to A, C to A, and D to B (see Figure 2). Emergency vehicles tend to use major routes wherever possible, so in our simulation they only travel on paths A to B, and B to A. Therefore, the EVO policy is only deployed on agents A, B, E, and F. All vehicles follow the shortest path from source to destination. Vehicle routes are the same for all of the experiments we ran.

Agent performance is tested for three different traffic loads to simulate different traffic conditions. The loads are as follows: low load (a total of 28,140 vehicles are inserted, 7,000 cars on each of the car routes and 70 ambulances on each of the emergency vehicle routes), medium load (a total of 56,280 vehicles, 14,000 cars on each of the car routes and 140 ambulances on each of the emergency vehicle routes) and high load (a total of 100,500 vehicles, 25,000 cars on each of the car routes and 250 ambulances on each of the emergency vehicle routes).

Each signalised junction in the simulation has a different set of available phases, automatically generated based on junction layout. For this set of experiments, the duration of each phase is set to 20 seconds. Junctions can cycle through their available phases using RR, or can be controlled by SAT or one of the RL agents described in the previous section.

4.5 Experiment Parameters

Each of our RL experiments is run in two parts: 2010 simulation minutes of exploration, and 2010 minutes of exploitation. The duration of 2000 minutes enables Q-learning to execute 6000 learning steps (as our actions are 20 seconds duration each) which, we consider sufficient for agents to learn the Q-values for their state-action pairs. Additional 10 minutes were added to allow a chance for last inserted vehicles to leave the system. GWEV-c has a much larger state space than the other policies and therefore was given a longer exploration phase of 20000 minutes to enable a larger portion of the state space to be visited a sufficient number of times. Each experiments is repeated three times, and average results from exploitation phase are presented in this paper.

Each RL process has been run multiple times to determine the best combination of α and γ . The final combinations used for the experiments presented are, for GWO: $\alpha = 0.1$ and $\gamma = 0.3$, for EVO: $\alpha = 0.9$ and $\gamma = 0.1$, for GWEV-c: $\alpha = 0.1$ and $\gamma = 0.1$ and for GWEV-w: $\alpha = 0.1$ and $\gamma = 0.7$.

The performance of SAT also varies based on the size of the steps in which the phase duration can be incremented or decremented, as well as the maximum duration of the cycle factor. The actual maximum duration of the cycle for a junction is a function of this factor and the number of available phases for that junction. The best parameters determined for SAT performance with a

minimum action duration of 20 seconds are 10 for the increment step, and 1.2 for the maximum duration of the cycle factor.

4.6 Results and Analysis

Metrics. We compared the performance of the RL agents based on the following metrics:

- Density - measured as the ratio of occupied road space to available road space [2]. For the same traffic arrival rate, higher density means worse agent performance, since traffic that is not successfully cleared and is still in the system is creating higher density.
- Waiting time - average waiting time per vehicle for the duration of the experiment. We separate waiting times per vehicle type, so we can measure performance towards each of individual policies described in Section 4.2.

Density. Density results are summarized in Table II.

Table 1. Average density per load level ratio

	RR	SAT	GWO	EVO	GWEV-c	GWEV-w
Low	2.96	2.76	1.66	12.30	1.60	1.49
Medium	5.60	5.20	3.31	11.37	3.47	3.09
High	11.04	9.50	5.84	15.85	6.03	5.06

We see that GWEV-w has the lowest density across all three loads, indicating that it is the most successful approach to managing the general traffic flow. GWO and GWEV-c have similar densities, with GWEV-c being better at the low load, and GWO at all other loads. We believe this is due to GWO addressing cars, which make up 99.5% of the total traffic, so its performance is very close to GWEV-c, which addresses all traffic. At the low loads, SAT and RR perform similarly, while the difference becomes more obvious at high load, where SAT performs better.

EVO has by far the highest density for all three loads. We believe this is due to the fact that this policy addresses only emergency vehicles, which make up only 0.5% of traffic in our simulation. Cars, which make up remaining 99.5% of the traffic, are not addressed, and create a backlog in the system. In Figure 3 we see an example of the effect of this backlog on the density. In the EVO implementation of the UTC system fills up with the traffic not adequately addressed by the policy, creating higher density and worse performance. This confirms the high dependency between policies due to the shared infrastructure, i.e. road space.

Emergency-vehicle waiting time. Our initial expectations were that EVO, whose only goal is to prioritize for emergency vehicles, would yield the best waiting times for emergency vehicles. However, this approach turned out to have the worst performance due to the high dependency between emergency vehicle performance and the performance of private vehicles which are not addressed by

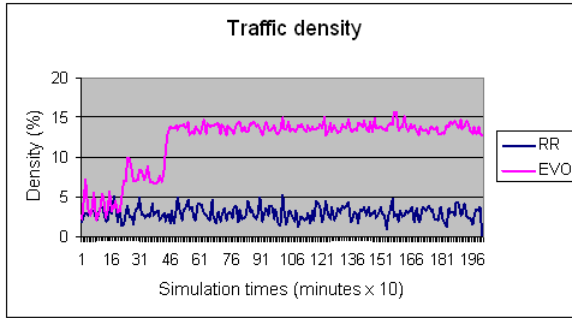


Fig. 3. Density during low load

this policy. Not only do emergency vehicles suffer high waiting times, but a large number of vehicles had to be turned away, as the system was backlogged and there was no available road space for them to join. For this reason, EVO waiting time results are not comparable to other results and we exclude them from subsequent graphs.

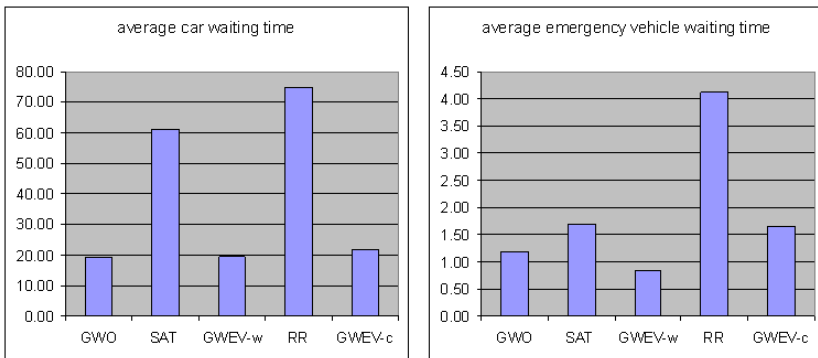


Fig. 4. Average waiting time per vehicle type

Figure 4 shows average waiting times for both cars and emergency vehicles for the medium traffic load for all the policies we implemented apart from EVO. For the moment, we focus on the emergency vehicle waiting times. On the graph shown, GWEV-w is the best policy for emergency vehicles, but at low and high loads, GWO slightly outperforms this policy. The similar performance of single-policy GWO and multi-policy GWEV-w suggests the high dependency between the performance of different vehicle types. It emphasises the importance of clearing general traffic, as GWO does, to free up the road space for emergency vehicles, and suggests a high dependency between a policy that addresses emergency vehicles and one that addresses private vehicles.

GWEV-w clearly outperforms GWEV-c at all loads, reducing emergency vehicle waiting time to between $\sim 40\%$ and $\sim 80\%$ of their waiting time in GWEV-c. We believe that worse performance of GWEV-c is caused by the size of the agents' state spaces. These results suggest that GWEV-c, even though it still outperforms SAT and RR, is not effective even for combinations of only two policies and would not be scalable to the addition of any further policies.

GWEV-w also outperforms our baselines, reducing average emergency vehicle waiting time to between $\sim 7\%$ and $\sim 50\%$ of their waiting time in SAT, and between $\sim 7\%$ and $\sim 20\%$ of their waiting time in RR. These results suggest that GWEV-w is a suitable technique for multi-policy optimization in UTC, as it outperforms the other evaluated multi-policy technique GWEV-c, single policy EVO, as well as both of our baselines.

Car waiting time. GWO, GWEV-w and GWEV-c have similar performance for all loads in terms of average car waiting time, with GWEV-w slightly outperforming GWEV-c, and GWO slightly outperforming both of the multi-policy approaches (see Figure 4). From this we conclude that the best waiting times for cars are achieved when the system optimizes only for cars (GWO), but in the presence of multiple policies, specifically one with a higher priority such as emergency vehicles, GWEV-w is the best approach. GWEV-w also outperforms our baselines, and reduces waiting time for cars to between $\sim 32\%$ and $\sim 42\%$ of their waiting time in SAT and between $\sim 26\%$ and $\sim 38\%$ of car waiting time in RR.

It is also interesting to observe the performance of our baselines, SAT and RR, in relation to each other, both in terms of car waiting time and emergency-vehicle waiting time. At the medium load (see Figure 4) and at the high load, the adaptive SAT algorithm performs better, as we expected, but at the low load RR actually performs better. This indicates that when the loads in the system are very low, running an adaptive algorithm, SAT, might have adverse effects on traffic performance, possibly due to extending phase times to longer than it is required and creating larger backlogs. However, these results also emphasize the importance of adaptation at higher loads.

Overall analysis. From the experiments we performed we have made following main observations. Both RL-based techniques, GWEV-w and GWEV-c, outperform our baselines, both in terms of emergency vehicle and car waiting times, showing that RL-based techniques are promising approaches to multi-policy optimization in autonomic systems. GWEV-w performs better than GWEV-c, indicating that W-learning-based approach is a more suitable approach for multi-policy optimization than combining learning processes into a single learning process. We also observe high dependency between the policies reflected in their performance. The policy that addresses only emergency vehicles (EVO) generates a backlog of other vehicles, and as a result performs very badly both in terms of car and emergency-vehicle waiting times. GWO, which addresses only cars, also performs well in terms of emergency vehicle waiting times, as clearing cars creates less congested roads and enables emergency vehicles to proceed. Our results also show that the importance of the optimization increases with the

traffic load, where the gap between the performance of adaptive techniques (e.g. SAT) and nonadaptive techniques (e.g. RR) grows larger.

5 Conclusions and Future Work

In this paper we presented the challenges of multi-policy optimization in decentralized autonomous systems. We have evaluated several proposed multi-policy optimization RL techniques in UTC and our results indicate that W-learning is a suitable approach for optimization towards multiple policies in multi-agent heterogeneous autonomous environments. In future work, we will extend the scope of our experiments to additional policies with different characteristics, to establish wider applicability of W-learning-based techniques. We will also investigate potential for performance improvement by agent collaboration by enabling W-learning agents to cooperate with each other in order to meet system goals.

Acknowledgements

This work was supported by Science Foundation Ireland grant 03/CE2/I303_1 to Lero - the Irish Software Engineering Research Centre (www.lero.ie). The authors would also like to thank As'ad Salkham for his work on the implementation of RL libraries used in our evaluation, and Vinny Reynolds, Anurag Garg, Raymond Cunningham, Mikhail Volkov, and Sylvain Cabrol for their work on the traffic simulator.

References

1. Abdulhai, B., Pringle, R., Karakoulas, G.: Reinforcement learning for the true adaptive traffic signal control. *Journal of Transportation Engineering* 129(3), 278–285 (2003)
2. Bazzan, A.L.: A distributed approach for coordination of traffic signal agents. *Autonomous Agents and Multi-Agent Systems* 10(1), 131–164 (2005)
3. Cuayáhuitl, H., Renals, S., Lemon, O., Shimodaira, H.: Learning multi-goal dialogue strategies using reinforcement learning with reduced state-action spaces. *Int. Journal of Game Theory*, 547–565 (2006)
4. Dowling, J.: The Decentralised Coordination of Self-Adaptive Components for Autonomous Distributed Systems. PhD thesis, Trinity College Dublin (2005)
5. He, L., Nort, N.: Hybrid genetic algorithms for telecommunications network back-up routeing. *BT Technology Journal* 18(4) (October 2000)
6. Humphrys, M.: Action Selection methods using Reinforcement Learning. PhD thesis, University of Cambridge (1996)
7. Kadrovach, B.A., Lamont, G.B.: A particle swarm model for swarm-based networked sensor systems. In: SAC, pp. 918–924 (2002)
8. Kephart, J.O., Chess, D.M.: The vision of autonomic computing. *Computer* 36(1), 41–50 (2003)
9. Meignan, D., Simonin, O., Koukam, A.: Simulation and evaluation of urban bus-networks using a multiagent approach. *Simulation Modelling Practice and Theory* 15(6), 659–671 (2007)

10. Montresor, A., Meling, H., Babaoğlu, Ö.: Messor: Load-balancing through a swarm of autonomous agents. In: Moro, G., Koubarakis, M. (eds.) AP2PC 2002. LNCS (LNAI), vol. 2530, pp. 125–137. Springer, Heidelberg (2003)
11. Natarajan, S., Tadepalli, P.: Dynamic preferences in multi-criteria reinforcement learning. In: ICML 2005: Proceedings of the 22nd international conference on Machine learning, pp. 601–608. ACM, New York (2005)
12. Oliveira, E., Duarte, N.: Making way for emergency vehicles. In: Proc. of the 2005 European Simulation and Modelling Conference, pp. 128–135 (2005)
13. Papageorgiou, M., Diakaki, C., Dinopoulou, V.: Review of road traffic control strategies. Proc. of the IEEE 91(12) (December 2003)
14. Pendrith, M.D.: Distributed reinforcement learning for a traffic engineering application. In: AGENTS 2000, pp. 404–411. ACM Press, New York (2000)
15. Reynolds, V., Cahill, V., Senart, A.: Requirements for an ubiquitous computing simulation and emulation environment. In: InterSense 2006. ACM Press, New York (2006)
16. Richter, S.: Learning traffic control - towards practical traffic control using policy gradients. Technical report, Albert-Ludwigs-Universität Freiburg (2006)
17. Russell, S., Norvig, P.: Artificial Intelligence - A Modern Approach. Prentice-Hall, Englewood Cliffs (2003)
18. Salkham, A., Cunningham, R., Garg, A., Cahill, V.: A collaborative reinforcement learning approach to urban traffic control optimization. In: International Conference on Intelligent Agent Technology (December 2008)
19. Shelton, C.R.: Balancing multiple sources of reward in reinforcement learning. In: Neural Information Processing Systems 2000, pp. 1082–1088 (2000)
20. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. A Bradford Book. MIT Press, Cambridge (2002)
21. Tesauro, G., Chess, D.M., Walsh, W.E., Das, R., Segal, A., Whalley, I., Kephart, J.O., White, S.R.: A multi-agent systems approach to autonomic computing. In: AAMAS 2004, pp. 464–471 (2004)
22. Wiering, M.: Multi-agent reinforcement learning for traffic light control. In: Proc. of 17th Int. Conf. on Machine Learning, pp. 1151–1158. Morgan Kaufmann, San Francisco (2000)

SACConf: Semantic Attestation of Software Configurations*

Hua Wang, Yao Guo, and Xiangqun Chen

Key Laboratory of High Confidence Software Technologies (Ministry of Education)
Institute of Software, School of EECS, Peking University, Beijing, China
{wanghua04, yaoguo, cherry}@sei.pku.edu.cn

Abstract. Remote attestation is one of the key functionalities provided by trusted platforms. Most current attestation approaches are based on cryptographic hash functions, which are appropriate to attest to relatively stable objects such as executables. However, they can not effectively deal with software configurations that could have many (or even infinite) trusted variants and could also be modified at run-time. This paper proposes SACConf, a novel semantic attestation approach to attesting to software configurations. SACConf uses a list of constraints to represent the challenger's trust policies, and verifies configurations based on semantic checks against the constraints, according to the semantic meanings of configurations rather than their hashes. An on-request measurement strategy is also added as a complement to the on-load strategy in order to capture potential modifications to configurations during execution. We implemented a prototype of SACConf and evaluations show that it could reduce the storage overhead from *exponential* to *linear* compared to hash-based approaches.

1 Introduction

In a distributed environment involving multiple platforms, the platforms could be owned and managed by different entities who might not trust each other. Platforms could also be compromised and running malicious code. Thus it is very important that a platform (*challenger*) is able to verify the software trust state of another platform (*attestor*).

Configuration is an important factor affecting software trust. Many programs can be tuned to behave in very different ways through user-specified configurations. Usually a program is not trusted if its configuration does not comply with the challenger's trust policy, even if its executable is launched correctly without being tampered with. Therefore attesting to configurations should be included in software attestation as well.

* This research is supported by the National High Technology 863 Program of China under Grant No. 2007AA01Z462 and 2008AA01Z133, the National Basic Research Program of China (973) under Grant No. 2009CB320703, and the Science Fund for Creative Research Groups of China under Grant No. 60821003.

The Trusted Computing Group (TCG) has developed a bottom-up measuring model and a hardware-based integrity-proving mechanism [21], based on which some attestation approaches have been proposed, such as TPod [8] and IMA [14]. These approaches are believed to be capable of reporting trust states more reliably than pure software approaches. Existing TCG-style approaches do not distinguish between executables and configurations. Both are proved using cryptographic hash functions and with the on-load measure strategy. That is, the attester measures executables and configurations by computing their hashes at load time, and reports their hashes to the challenger during attestation. The challenger verifies these hashes by comparing them with pre-stored, trusted hashes.

Although TCG-style approaches may be suitable for executables, they do not work well for configurations due to the following two reasons. 1) When dealing with configurations, the hash space could explode very easily. Unlike executables that do not have many trusted variants, the number of trusted configurations could be extremely large or even infinite, for example, when considering a configuration entry that accepts a float number within a given range. Since with TCG-style approaches each configuration is denoted by a unique hash, it is sometimes impractical or impossible to deal with all trusted hashes, such as storing all of them. 2) The on-load measurement strategy can not always reflect the latest configuration. While executables usually do not change after being loaded, configurations of programs, such as *Firefox*, could be modified on the fly. In such cases the measurement performed at load time can not accurately indicate the trust state.

Some approaches have been proposed to mitigate the problem of hash explosion. Virtual machine (VM) based approaches such as Terra [3] separate trusted VMs and normal VMs, and only verify the hashes of programs running in the trusted VMs. Property-based attestation [12, 11, 1] introduces a trusted third party to examine the attester's hashes and returns its properties rather than hashes to the challenger. PRIMA [5] only attests to programs having information flow to trusted objects. These approaches, however, are still based on hash functions and focused mainly on executables.

The main limitation of hash-based approaches is that they do not take the internal structural information of configurations into account because this information can not be carried by hash values. Hence semantic checks such as range comparison and pattern matching can not be performed, thus the trust policy can only be represented by enumerating all trusted hashes.

To address this problem, we propose a new attestation approach, called Semantic Attestation of Configuration (SACConf), which represents the challenger's trust policy in a semantic way (i.e. using a group of constraints), and verifies configurations against these constraints according to the internal contents of configurations rather than their hashes. The verification is based on semantic checks, so the challenger could use a small list of constraints to match a large number of configurations. To protect the attester's privacy, the configuration is not sent to the challenger. Instead, the challenger sends its trust policy to the attester, and the attester is responsible for verifying whether its configuration

complies the policy and reporting the result. To assure the challenger of the genuineness of the attestation result, the attester also proves that the measuring and verifying code is executed correctly.

We add an on-request measurement strategy to SACConf as a complement to the on-load strategy. With this strategy the attester performs configuration measurement each time when receiving an attestation request from the challenger, so that the measurement result always reflects the current state at the moment when the request is processed. We implemented a prototype of SACConf to demonstrate its feasibility, in which we develop an example trust policy representation scheme for entry-based configurations, reducing the storage cost from exponential to linear. The time cost may increase with the on-request strategy, but experiment results show it is trivial.

The rest of this paper is organized as follows. Section 2 describes the design of SACConf. Section 3 describes the implementation and evaluation of the prototype. Section 4 discusses several additional advantages of SACConf. The related work is discussed in section 5. Finally we draw a conclusion and present our future work in section 6.

2 Semantic Attestation

TCG-style attestation approaches are based on cryptographic hash functions, requiring the challenger to store one hash for each trusted file. For executables, this method works well, since the number of trusted variants of an executable is not very big, usually including the original version and a number of patched versions. However, the number of trusted configurations could be extremely large, sometimes even infinite. For example, in terms of entry-based configurations consisting of $\langle \textit{entry}, \textit{value} \rangle$ pairs, a challenger may accept more than one value as trusted for some configuration entries. The values of these entries could be combined to produce a tremendous amount of trusted configurations. Worse still, some entries may even have unlimited choices. For example, it is not odd that the challenger does not care about the name of the user who runs Apache’s *httpd*, as long as its privileges are properly set. In this case the *User* entry of *httpd*’s configuration could be assigned arbitrary names, leading to actually countless trusted configurations. Consequently, it is often impractical or even impossible for the challenger to store hashes of all trusted configurations.

To attest to configurations which have internal structures and can be parsed according to their syntax, a better way is to read their contents and perform semantic checks on them to determine whether they comply with the trust policy. A trust policy usually consists of a group of constraints, such as “*the server at least supports hmac-sha1 algorithm*”, which must be satisfied by trusted configurations. We can represent the constraints in a formal way, so that semantic checks can be done automatically.

Semantic checks require SACConf to understand syntax of configurations. Hence some operations definitely depend on the configuration syntax of programs to be attested, including representing the challenger’s trust policy, and measuring

and checking the configuration. Configurations of different programs could be in very diverse forms, such as entry-based, rule-based and command-based configurations. It is hard to develop a common solution for all programs. Alternatively, we develop a flexible framework in which syntax-dependent operations are encapsulated in customizable and replaceable components.

Semantic checks also require that the party who performs the checks knows both the configuration and the trust policy. There are several candidates. A straightforward method is that the attestor sends the entire configuration to the challenger, which in turn checks the configuration against its policy. This approach, however, exposes the attestor's privacy to the challenger. A more sophisticated method is that both the configuration and the trust policy are sent to a reliable third party, which performs the check on behalf of the challenger. But this method requires the presence of an extra third party that is not always available. The method adopted by SACConf is that the challenger sends its policy to the attestor and the latter is responsible for performing the check. This method eliminates the above two deficiencies.

Because the configuration measurement and checks are all done in the attestor side, the attestor should provide necessary proof to convince the challenger of the genuineness of the attestation result. The facts to be proved include that the components of SACConf are not compromised and the trust policy and attestation result are not tampered with. To do so SACConf provides the attestor with a proving mechanism built on TCG's technology.

3 Design of SACConf

This section describes the design of SACConf, including the framework and the attestation process, as well as the mechanism for proving the genuineness of attestation results.

3.1 Framework of SACConf

The framework of SACConf is depicted in Figure 1, where syntax-dependent components are denoted by grey boxes, while syntax-independent components are denoted by white boxes.

The challenger needs to store its *trust policy* and send it to the attestor during attestation. Hence SACConf should provide the challenger with schemes to represent its policies. It is impractical to design a common scheme for all programs, but we could improve the flexibility of schemes so that they can be shared by a group of programs with similar configuration syntax. In next section we will present an example scheme which can be used for most entry-based configurations.

The *Attestation Server (AttServ)* runs as a daemon in the attestor. It is responsible for handling the interaction with the challenger and coordinating the activities of other components within the attestor.

The *Measurement Engine (MEngine)* is responsible for measuring configurations. Programs with distinct configuration syntax usually require customized

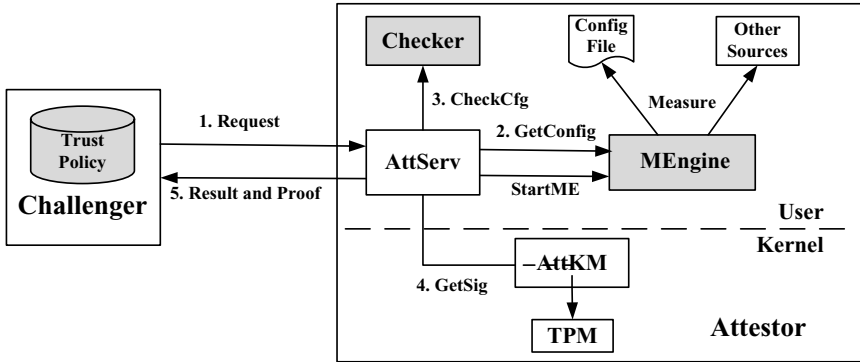


Fig. 1. SACon Framework

MEngines. The relationship between programs and MEngines is registered in a dedicated file.

The *Checker* performs semantic checks, examining whether the configuration measured by the MEngine satisfies the trust policy received from the challenger. The checking method depends on the policy representation scheme.

The *Attestation Kernel Module (AttKM)* is a kernel module used to invoke TPM to generate signatures to prove the genuineness of the attestation result. It also monitors execution of programs and notifies the AttServ of this information.

3.2 Attestation Process

The configuration attestation is accomplished through the collaboration of the components. The numbers in Figure 1 stand for the temporal order.

In terms of configuration measurement, most existing attestation approaches use the on-load strategy, i.e. measuring the configuration when it is loaded. We propose adding an on-request strategy, i.e. measuring the configuration when an attestation request is processed. SAConf supports both strategies. The former is suitable for programs whose configurations do not change on the fly, such as *sshd* and *httpd*; while, for programs whose configurations may be modified dynamically, e.g. *Firefox*, the latter is desired. Which strategy is used can be configured for each program.

Measurement is performed by MEngines launched by the AttServ. If a program is configured to use the on-load strategy, its MEngine is launched when the program is executed. Monitoring execution of programs is done by the AttKM, which hooks the kernel function used to execute programs, e.g. *do_execve* in Linux, and signals the AttServ if execution is detected. If the on-request strategy is used, the MEngine is launched after the AttServ receives a request from the challenger. In both cases the AttServ calculates the hashes of MEngines when launching them, in order to prove their integrity.

If the on-load strategy is used, the measurement result, as well as the hash of the MEngine, needs to be stored and used for attestation requests arriving

later. We use a dedicated Platform Configuration Register (PCR) [21] to protect both of them. While, with the on-request strategy, the result and the hash are consumed immediately after measurement and only for the current request, so it is not necessary to store them.

The attestation process varies slightly when using different strategies. The detail of each step is described as follows.

1. The attestation begins with the challenger sending a request to the AttServ, which contains the identity of the program to be attested and the trust policy for it, as well as some assistant data such as *nonce*, a random number used to defend against replay attacks.
2. Upon receipt of the request, the AttServ retrieves the program's identity and finds out which measurement strategy is configured for the program. If the on-load strategy is used, the AttServ fetches the configuration measured at load time, as well as the hash of the MEngine, and examines their integrity according to the PCR. Otherwise the AttServ looks up the MEngine registered for the program, and launches it to measure the current configuration, with the hash of the MEngine being computed.
3. The AttServ then launches a proper Checker, passing the trust policy and the configuration to it. The Checker examines whether the policy is satisfied by the configuration based on semantic checks. Like the MEngine, the Checker's hash is also calculated, used to prove its integrity.
4. The AttServ requests the TPM to generate a signature through the system call provided by the AttKM, used as the evidence of the genuineness of the attestation result. The computation of the signature involves the attestation result, the trust policy, and the hashes of SACConf's components.
5. The result, the hashes and the signature are sent back to the challenger. The challenger first verifies the integrity of the trust policy, the result and the hashes according to the signature. Then it verifies according to the hashes whether the SACConf's components are valid and launched correctly. If so, the challenger is assured that the result denotes the actual trust state of the program's configuration.

The attested program does not participate in the attestation process. There is also no synchronization requirement between the components of SACConf and the program. So applying SACConf does not need to modify existing programs.

3.3 Proving the Genuineness of Attestation Results

SACConf provides the attester with a proving mechanism to vouch for the genuineness of the attestation result. This mechanism is built upon TCG's technology, based on the assumption that a traditional TCG-style attestation approach, such as [8, 9, 14], has been implemented in the attester side.

TCG-style attestation approaches measure all loaded programs and modules, and the measurement results are categorized and stored in separate sequences, such as the sequences for the kernel and for applications, with corresponding

PCRs being extended. These sequences, as well as PCRs, are sent to the challenger during attestation. The challenger verifies the integrity of these sequences by calculating their hashes and comparing them with corresponding PCRs.

We use the traditional approach to prove low-level software from BIOS up to the kernel, including the AttKM. But the rest of SAConf’s components, which are user-mode programs, are not proved by the traditional approach, due to the following two reasons. 1) MEngines and Checkers should be not only legal but also proper. Here the term “legal” denotes that a program has been certified by a trusted third party, while “proper” denotes that a program works correctly in a certain situation. A legal program may be not proper if used in a wrong place. For example, the MEngines for *sshd* and *httpd* are all legal, but using the former to measure *httpd* is obviously not proper. Traditional approaches can not detect the improper use of legal programs. 2) When verifying configurations, the challenger usually only concerns SAConf’s components. But with traditional approaches, the challenger has to process the whole sequence containing the hashes of all loaded applications. The cost of this operation is high, especially when the system has been running for a long period and a lot of loads have occurred.

We propose a customized, more lightweight mechanism to prove the user-mode components. For the AttKM has been proved by the traditional approach, we use it to prove the AttServ, which in turn is used to prove Checkers and MEngines. The AttServ measures Checkers and MEngines when launching them. The AttServ itself is measured by the AttKM when it invokes the system call provided by the AttKM to generate signatures. The measurement results, i.e. hashes, are sent back to the challenger along with the attestation result, so the challenger can tell which MEngine and Checker are used.

The AttKM invokes the TPM to generate a signature to prove the hashes of SAConf’s components. Because only the kernel is allowed to access the TPM, this signature can not be forged by malicious applications. Besides, in order to prove the integrity of the trust policy and the attestation result, their hashes are also included in the computation of the signature.

Finally, what the challenger receives from the attestor is

$$\langle result, hs, sig \rangle$$

where *hs* and *sig* are the proof. *hs* is the hashes of the Checker, the MEngine and the AttServ. *sig* is the signature generated by the TPM whose value is

$$sig = S_K(H(result), H(tp), H(hs), nonce)$$

where $S()$ and K are the signing function and key used by the TPM; $H()$ is a hash function such as SHA1; *tp* is the challenger’s trust policy; and *nonce* is a random number generated by the challenger to prevent replay attacks.

When receiving the package, the challenger retrieves the attestation result, *hs* and *sig*. It also knows *tp*, *nonce* and the public part of K . Then it verifies the genuineness and freshness of the package by examining the signature and *nonce*,

followed by verifying the hashes of SACConf’s components. If all verification is passed, the challenger believes that the attestation result reflects the actual trust state of the program’s configuration.

4 Implementation

We implemented a prototype of SACConf in a Dell OptiPlex 620 equipped with a TPM chip, running Linux 2.6.20. This section describes its implementation and evaluation.

4.1 Representation Scheme for Entry-Based Configurations

Although developing a thoroughly common trust policy representation scheme for all programs is infeasible, we could develop flexible schemes that can be used for a large number of programs. The entry-based configuration is one of the most widely used configuration forms. We develop an example scheme that can be used for most of this kind of configurations.

For entry-based configurations, a challenger’s trust policy are typically some logical conditions each of which must be satisfied by the values of configuration entries. For example, a challenger requires the value of an entry must be greater than a constant. Straightforwardly, we use a set of boolean expressions to represent these conditions. Each expression involves one or several configuration entries. If their values make the expression true, the expression is said to be satisfied. If all expressions are satisfied, the configuration of the program is considered to be compliant with the challenger’s policy.

The boolean expression is somewhat similar to that of high-level languages. Most of often-used operators are supported in our scheme. For entries with numeric values, all arithmetic, relational and logical operators are supported. For entries with string values, “=” and “!=” are supported, as well as some frequently used string functions such as *strcmp*, *strlen* and *strstr*. Entry values are referred to by the $\$()$ operator. $\$(entryid)$ will be replaced by the value of the entry specified by *entryid* when the expression is evaluated.

In addition, we add two extensions to enhance the representation capability of our scheme. The first extension is to support set operators. In some cases it is convenient for the challenger to treat some entries (e.g. lists) as sets. For example, the challenger may require that its ID is in the trusted ID list. This can be represented by a *belong* operator easily. Supported set operators are listed in Table 1. Two special sets are defined. The empty set is denoted by Φ and the universe is denoted by Ψ . The second extension is to support regular expressions. Regular expressions can be used to specify entry IDs in the $\$()$ operator, or describe patterns that the values of trusted entries must match.

An example trust policy for *sshd* represented by our scheme is shown in Figure 2. Its meaning is straightforward.

Our scheme enables semantic checks such as range comparison and regular expression, which are not supported by hash based approaches. With this scheme

Table 1. Supported Set Operators

Operator	Syntax	Description
set	$\text{set}(d,L)$	This operator is used to convert a list to a set. L is the string that denotes the list, and d is the delimiter that separates elements in the string.
==	$S_1 == S_2$	If S_1 contains the same elements as S_2 , return <i>true</i> ; otherwise return <i>false</i> .
!=	$S_1 != S_2$	Return the reverse result of "==".
belong	$e \text{ belong } S$	If element e belongs to S , return <i>true</i> ; otherwise return <i>false</i> .
incl	$S_1 \text{ incl } S_2$	This is the inclusion operator. If S_1 includes S_2 , return <i>true</i> ; otherwise return <i>false</i> .
union	$S_1 \text{ union } S_2$	This operator returns the union of S_1 and S_2 .
inters	$S_1 \text{ inters } S_2$	This operator returns the intersection of S_1 and S_2 .
diff	$S_1 \text{ diff } S_2$	This operator returns the difference of S_1 and S_2 .

```
[sshd, f8e3e1cd58fdb8434497c0c9ca783bc3cf6a38b3]
// The supported protocol version must be included in set {1,2}
#1 set(,"1,2") incl set(, $(Protocol))

// root is not allowed to log in through ssh
#2 !("root" belong set(, $(AllowUsers))) || ("root" belong set(, $(DenyUsers))) \
|| ($(PermitRootLogin) == "no")

// The intersection of AllowUsers and DenyUsers should be a empty set
#3 (set(, $(AllowUsers)) inters set(, $(DenyUsers))) ==  $\emptyset$ 

// the cvs group is allowed to log in
#4 "cvs" belong set(, $(AllowGroup))

// expressions for password authentication
#5 $(PasswordAuthentication) == "yes"
#6 $(PermitEmptyPasswords) == "no"
#7 $(UsePAM) == "yes"
#8 $(MaxAuthTries) <= 6
#9 $(MaxStartups) > 5 && $ < 10
```

Fig. 2. An Example Trust Policy for *sshd*

the challenger can use a group of boolean expressions to match a large number of configurations, so its storage overhead decreases significantly.

4.2 Evaluation

In this subsection we will evaluate the storage and time cost of SACConf respectively.

Storage Cost. Our trust policy representation scheme reduces the storage complexity for the challenger from *exponential* to *linear*.

With hash-based approaches the challenger needs to store one hash for each trusted configuration. The number of trusted configurations can be computed approximately as follows. Suppose there are n entries in the configuration and

Table 2. Time Cost

	Measuring	Checking	Reporting	Responding
SACConf(On-request)	544 μ s	9 μ s	932751 μ s	933304 μ s
SACConf(On-load)	24166 μ s	135 μ s	931773 μ s	931908 μ s
IMA	23394 μ s	-	927168 μ s	927168 μ s

the number of trusted values for the i th entry is c_i . Then the number of trusted configurations is $\prod_{i=1}^n c_i$. So the storage cost grows *exponentially* with the number of entries having multiple trusted values. Here we do not take factors such as comments and the order of entries into account, otherwise the cost will be infinite.

While with our scheme, the challenger only needs to store some boolean expressions. So the storage cost grows *linearly* with the number of the constraints, which is usually less than the number of entries.

Time Cost. We measured the time cost of SACConf and compared it with that of IMA. The result is shown in Table 2. The operations done in the attester side can be divided into three stages: measuring the configuration, checking the configuration, and reporting the result. Their cost is listed in corresponding columns.

Each stage is comprised of several operations, some of which are accomplished by invoking TPM commands. Because of the limited computational ability of TPM, the time spent by these commands accounts for the majority of the cost.

When using the on-load measurement strategy, the measurement is only performed at load time, but SACConf needs to extend PCRs for each measurement, which is similar to IMA, making its measurement cost close to that of IMA. When using the on-request strategy, the measurement is only used for the current attestation request. SACConf does not need to store the result and extend PCRs to protect it, so the cost is much lower.

The checking stage is only necessary for SACConf. With the on-load strategy, after retrieving the previously measured and stored result, SACConf still needs to get the specific PCR and verify the result accordingly; while this operation is not necessary when using the on-request strategy. Therefore the cost of the latter is much lower than that of the former.

With respect to the reporting stage, the cost of SACConf and IMA is close. In the reporting stage, SACConf needs to invoke the TPM to generate a signature as evidence, while IMA needs to get a signed PCR from the TPM. Both need the signing operation of the TPM, which contributes to more than 99% of the cost of this stage.

The last column shows the responding time, namely the time from an attestation request being received to the final result for the request being returned. For SACConf with the on-request strategy, the responding time includes the time spent in all stages; for SACConf with the on-load strategy, it includes the checking and reporting stages; while for IMA, it only contains the reporting stages. But from the result we can see that there is no big difference between their responding time.

Consequently, although some operations of SAConf, such as measuring for each attestation request, parsing configuration files and checking semantically, may cause extra time cost, it has little compact on the total performance because the majority of time overhead is spent by the TPM.

5 Related Work

Establishing trust among platforms is an important requirement in distributed environment. This is usually done by the attestation technologies. There have already been a lot of attestation approaches proposed.

Some approaches are based on secure hardware. TCG has developed TPM [20], a secure chip that has been shipped with many platforms. Based on TPM TCG proposed a hash-based integrity-proving mechanism [21]. The hashes of files, including executables and configuration files, are computed at the load time, and reported to the remote platform later as the trust evidence.

TCG's proving mechanism has been widely adopted. TPod [8] implements extensions to the grub bootloader to measure the sequence of code loads that bring up the operating system, and it stores these measurements in the TPM to protect them from tampering by software. The TPM can create signed messages that enable a remote party to verify the code loads measured by TPod. IMA [14] steps further by extending integrity measurement and verification up to the application level. Sailer et al. limit clients' access to the corporate network according to their integrity property [13].

Some researches attempt to make improvements to TCG's mechanism. Property-based attestation [12, 11, 1] concerns with the platform's property, such as security level, rather than the hashes of loaded files. A trusted third party is introduced to map hashes to platform's properties. Through the property-based attestation, [11] intends to protect the attestor's privacy. Instead, in SACon we use the privacy policy mechanism to do so.

Semantic remote attestation [4] attempts to attest to the program's behavior, rather than the integrity of its executable and configuration file. The approach is based on language-based trusted virtual machines (VM). It utilizes the high-level semantic information contained in the portable code to deduce the program's behavior. Although we all use the word "semantic", it has different meanings. In semantic remote attestation it means the information in executables, while in SACon it denotes the information in configuration files.

Attesting to executables will also encounter the problem of multiple versions, though this problem is not so serious as that of configuration files. Some researches have identified and tried to solve this problem. Terra [3] uses a VM based approach, which provides trusted VMs and normal VMs simultaneously, running high security-level and low-security level applications separately. The challenger only needs to verify the software stack in trusted VMs, so the hashes that the challenger needs to store are reduced. PRIMA [5] extends the IMA by coupling IMA to SELinux policy [7]. The number of measurement targets is reduced to those that have information flows to trusted objects. BIND [19]

allows programmers to mark out the critical code region through an attestation annotation mechanism. Only the hashes of the critical regions are computed and reported. These regions are more stable than other code and data regions. However, all these approaches focus on the attestation of executables. None of them can solve the problem of configurations effectively.

The attestation result can only prove the transient trust state, but not persistent state. BIND [19] attempts to prolong the trust state by moving attested code into a sand-boxing to protect its execution. However BIND only concerns with code, but not configurations. SACon mitigates this problem by providing the challenger with the capability of requesting the attestation at any moment. The challenger can send requests periodically or at random time.

In addition to hardware based approaches, there have also been some pure software approaches proposed. Genuinity [6] explores the problem of detecting the difference between a simulator-based computer and an actual computer. Genuinity relies on the premise that the program execution based on simulator is bound to be slower than that based on the actual computer. The execution time of a specific function that computes the checksum of memory is used as the evidence to make the attestation. However, Shankar et al. shows that side effects are not enough to make the attestation [18]. Pioneer [15] also relies on the execution time to perform the attestation. A deliberately designed verification function and a challenger-response protocol are used to establish the dynamic root of trust, which can ensure the succeeding code execution. TEAS [2] sends a randomly selected code segment, rather than a fixed function, to the remote platform, and determines the genuineness of the remote platform according to the returned result and the consumed time.

SWATT [16] is a technique proposed to perform attestation on embedded device with simple CPU architecture. It uses a well-constructed verification function so that any attempt to tamper with it will increase the running time. There are still some other attestation approaches proposed for embedded systems [10, 17, 22].

6 Conclusion and Future Work

Trust attestation of a software system must take its configuration into account. Existing attestation approaches can not attest to the software configuration effectively because of high storage overhead and inability to prove the latest states of configurations that can be modified dynamically.

In this paper we propose SACConf, a semantic attestation approach, to overcome these problems. The key contribution of SACConf is that it integrates semantic checks with TCG's hardware-based proving mechanism. The genuineness of the checks is proved by a TPM-based mechanism. The challenger's trust policies are also represented semantically, reducing the space complexity from exponential to linear. Besides, we introduce an on-request measurement strategy, which can accurately reflect the trust state at the moment when the request is processed.

We demonstrate the feasibility of SACConf by implementing a prototype. Experiments show that SACConf could reduce the storage overhead significantly compared to state-of-the-art approaches, only with a very little time cost increase.

Currently we have developed a trust policy representation scheme for entry-based configurations. Schemes for other configuration forms, such as rule-based and command-based configurations, still need to be developed, and could be very different from this one. In future we will study these configuration forms and develop proper schemes for them.

References

- [1] Chen, L., Landfermann, R., Lohr, H., Rohe, M., Sadeghi, A.-R., Stable, C.: A Protocol for Property-Based Attestation. In: *The 1st ACM Workshop on Scalable Trusted Computing*, Alexandria, Virginia, USA, pp. 7–16. ACM, New York (2006)
- [2] Garay, J.A., Huelsbergen, L.: Software Integrity Protection Using Timed Executable Agents. In: *The 2006 ACM Symposium on Information, Computer and Communications Security*, Taipei, Taiwan, pp. 189–200 (2006)
- [3] Garfinkel, T., Pfaff, B., Chow, J., Rosenblum, M., Boneh, D.: Terra: A Virtual Machine-Based Platform for Trusted Computing. In: *The 19th Symposium on Operating System Principles*, Bolton Landing, New York, USA, pp. 193–206 (2003)
- [4] Haldar, V., Chandra, D., Franz, M.: Semantic Remote Attestation - A Virtual Machine Directed Approach to Trusted Computing. In: *The Third Usenix Virtual Machine Research and Technology Symposium*, San Jose, CA, USA, pp. 29–41 (2004)
- [5] Jaeger, T., Sailer, R., Shankar, U.: PRIMA: Policy-Reduced Integrity Measurement Architecture. In: *The 11th ACM Symposium on Access Control Models and Technologies*, Lake Tahoe, California, USA, pp. 19–28. ACM Press, New York (2006)
- [6] Kennell, R., Jamieson, L.H.: Establishing the Genuinity of Remote Computer Systems. In: *The 12th USENIX Security Symposium*, Washington, DC, USA, pp. 295–308 (2003)
- [7] Loscocco, P., Smalley, S.: Integrating Flexible Support for Security Policies into the Linux Operating System. In: *FREENIX Track: 2001 USENIX Annual Technical Conference*, Boston, Massachusetts, USA, pp. 29–42 (2001)
- [8] Maruyama, H., Seliger, F., Nagaratnam, N., Ebringer, T., Munetoh, S., Yoshihama, S., Nakamura, T.: Trusted Platform on Demand. Technical Report RT0564, IBM (February 2004)
- [9] Microsoft. Secure Startup - Full Volume Encryption: Technical Overview (April 2005)
- [10] Park, T., Shin, K.G.: Soft Tamper-Proofing via Program Integrity Verification in Wireless Sensor Networks. *IEEE Transactions on Mobile Computing* 4(3), 297–309 (2005)
- [11] Poritz, J., Schunter, M., Van Herreweghen, E., Waidner, M.: Property Attestation - Scalable and Privacy-friendly Security Assessment of Peer Computers. Technical Report RZ 3548, IBM Zurich Research Laboratory (October 2004)
- [12] Sadeghi, A.-R., Stubble, C.: Property-based Attestation For Computing Platforms: Caring about Properties, Not Mechanisms. In: *The 2004 workshop on New Security Paradigms*, Nova Scotia, Canada, pp. 67–77 (2004)

- [13] Sailer, R., Jaeger, T., Zhang, X., van Doorn, L.: Attestation-based Policy Enforcement for Remote Access. In: The 11th ACM Conference on Computer and Communications Security, Washington, DC, USA, pp. 308–317. ACM Press, New York (2004)
- [14] Sailer, R., Zhang, X., Jaeger, T., van Doorn, L.: Design and Implementation of a TCG-based Integrity Measurement Architecture. In: 13th USENIX Security Symposium, San Diego, California, pp. 223–238 (2004)
- [15] Seshadri, A., Luk, M., Shi, E., Perrig, A., van Doorn, L., Khosla, P.: Pioneer: Verifying Code Integrity and Enforcing Untampered Code Execution on Legacy Systems. In: Advances in Information Security, vol. 27, pp. 253–289. Springer, US (2005)
- [16] Seshadri, A., Perrig, A., van Doorn, L., Khosla, P.: SWATT: SoftWare-based ATTestation for Embedded Devices. In: The 2004 Symposium on Security and Privacy, pp. 272–282 (2004)
- [17] Shaneck, M., Mahadevan, K., Kher, V., Kim, Y.: Remote Software-Based Attestation for Wireless Sensors. In: Molva, R., Tsudik, G., Westhoff, D. (eds.) ESAS 2005. LNCS, vol. 3813, pp. 27–41. Springer, Heidelberg (2005)
- [18] Shankar, U., Chew, M., Tygar, J.D.: Side Effects Are Not Sufficient to Authenticate Software. In: The 13th USENIX Security Symposium, pp. 89–102 (2004)
- [19] Shi, E., Perrig, A., Van Doorn, L.: BIND: A Fine-grained Attestation Service for Secure Distributed Systems. In: IEEE Symposium on Security and Privacy, pp. 154–168 (2005)
- [20] TCG. TPM Main Part 1 Design Principles (March 2006)
- [21] TCG. TCG Specification Architecture Overview (August 2007)
- [22] Yang, Y., Wang, X., Zhu, S., Cao, G.: Distributed Software-based Attestation for Node Compromise Detection in Sensor Networks. In: The 26th IEEE International Symposium on Reliable Distributed Systems, pp. 219–228. IEEE Computer Society, Los Alamitos (2007)

ALOPA: Authorization Logic for Property Attestation in Trusted Platforms

Aarthi Nagarajan, Vijay Varadharajan, and Michael Hitchens

Macquarie University, Sydney, Australia
{aarthi, vijay, michaelh}@science.mq.edu.au

Abstract. Property based attestation is an extension of the proposed trusted computing attestation mechanism where binary measurements are abstracted to meaningful platform properties. In this paper, we propose ALOPA - Authorization Logic for Property Attestation, a logic based language for the specification and evaluation of authorization policies using properties in trusted platforms. Access control policies specified using ALOPA govern the access of platforms to resources on the basis of the platform's identity and a collection of rules based on platform properties, which determine, for any platform and any resource, the types of accesses the platform is allowed on the resource. Such an approach seems promising for developing secure distributed applications using property attestation based authorization for trusted platforms.

1 Introduction

An important requirement for information security is the protection of data and resources from unauthorized disclosure and modification, while ensuring that legitimate users have access to authorized resources at all times. In systems where information is distributed, when one entity requests a resource from another, the receiving entity needs to address at least two fundamental questions. How does one believe that the requesting entity is the one that it claims to be and does the requesting entity have appropriate privileges for the requested service? These two questions relate to the issues of authentication and authorization. The authorization requirements in distributed applications are much richer than that of authentication both in terms of the types of privileges required and the nature and degree of interactions between the participating entities. Over the years, many different authorization systems have been developed to suit different access control requirements.

Typically, authorization systems use attributes or credentials associated with a user (human being) to control access to resources. For example, passwords, certificates and bio-metric information is used to prove that the necessary privileges are held for a request to be serviced. If the system is convinced that the user is legitimate, it will allow access to the requested resources. Such authorization systems however make a fundamental assumption that the underlying computing platform of the user is 'secure and trusted'. The authorization system assumes that the hardware, OS and all applications running on the user's system are safe and will not cause any harm to it or its system and resources once the user has been authorized. In some sense, the user and the user's platform are considered as one single entity for authorization. In the current networked

world, such assumptions are clearly inappropriate because computing platforms are becoming more and more susceptible to changes that are outside the control or knowledge of its (legitimate) user due to malicious software. The challenge is then, how can one reason about the identity, integrity and security of a platform in the distributed world when platforms are vulnerable to dynamic changes affecting their behaviour.

One recent advancement to address this issue has been the initiative from the Trusted Computing Group or TCG. The stated goal of trusted computing technology [1] is to improve the security and trustworthiness of computing platforms. It provides mechanisms to perform certain cryptographic functions and to protect secret keys and data of the platform. Perhaps, the most important feature of trusted computing is its ability to validate the configuration of a platform [2]. It has special processes that securely collect and report information about the correctness of the system configuration in order to prove its validity to a third party. Using the reported information, the third party may decide whether it wants to continue communicating with the platform or not. This can be considered as the first step to authorizing computing platforms. In this paper, we present an authorization language, ALOPA - Authorization Logic for Property Attestation that can be used to specify and evaluate authorization policies taking into account the state of the platform. Policies specified using ALOPA govern the access of platforms to resources on the basis of the platform's identity and a collection of rules (based on platform properties) which determine, for any platform and any resource, the types of accesses the platform is allowed on the resource. ALOPA is simple yet powerful enough to capture all access control requirements based on the components, properties of components and their relationships in a given platform.

The rest of the paper is organized as follows. Section 2 provides a brief overview of some of the basic concepts in trusted computing platforms and attestation. Section 3 describes the ALOPA language in detail. Section 4 outlines the authorization derivation mechanism along with two algorithms for policy resolution and evaluation. Section 5 considers a small application scenario to illustrate the use of ALOPA. Finally, the paper concludes in Section 6.

2 Trusted Computing and Attestation

Trusted computing technology aims towards providing techniques for achieving security using hardware in computing platforms. Trusted platforms can securely store secrets and data using a special chip called the Trusted Platform Module (TPM) that is embedded in the motherboard at the time of platform manufacture. They can also collect and provide evidence on the state of the hardware, firmware and software that are installed on the platform. When a trusted platform boots, all processes starting from the boot measure the next process to be loaded. All measurements are in the form of a 160 bit hash that are stored inside special registers called the Platform Configuration Registers (PCR) within the TPM. A log of the measurements is also stored outside the TPM in local storage. When a third party wishes to learn the state of a trusted platform, it initiates a process known as 'attestation'. During attestation, the TPM collates the requested PCR values, the measurement log and the reference measurements for each component measured into an Integrity Report (IR). When the host receives the

report, it validates each individual measurement using the log and the reference values to determine if the components inside the platform are in the expected state.

Recently, TCG attestation technique has become a much debated topic. Researchers believe that measured hash values are only a representation of the program implementation rather than the program properties which are more important for security. This will favour certain implementations (like paid software to open source) and not others even though both may provide similar security features. Also, hash values are cumbersome and are only machine understandable which makes their usage in security policies hard. For example, if a software application has one security property and dozens of hash values that correspond to a software state that guarantees that property, then it is easier to write security policies using the property rather than the hash values. This led to the proposal of the property based attestation [2] [3] [4] mechanism as an extension to the attestation technique. Using property based attestation, it is possible to prove that the availability of a certain hash measurement guarantees the availability of a certain security property thereby abstracting low level binary values to more meaningful attributes.

Property based attestation may be realized using different design choices. In this paper, we use the following simple protocol. Initially, a manufacturer produces a software component \mathcal{C} for a trusted platform \mathcal{TP} and provides its expected hash measurement \mathcal{H} as a reference value. Then a certification authority \mathcal{CA} certifies that the component \mathcal{C} with hash \mathcal{H} has certain property \mathcal{P} . This certificate is denoted by $Cert_{\mathcal{CA}} := (\mathcal{C}, \mathcal{H}, \mathcal{P})$. Sometime later, an attestation requester wishes to learn what security properties are present in \mathcal{C} before it can start communicating with \mathcal{TP} . \mathcal{TP} provides the attestation requester with its Integrity Report (IR) and $Cert_{\mathcal{CA}}$. The attestation requester first verifies if the platform measurements match the manufacturer's reference measurements \mathcal{H} . If so, the requester concludes that \mathcal{C} has property \mathcal{P} as certified in $Cert_{\mathcal{CA}}$. If not, it may not communicate with \mathcal{TP} . A similar but slightly modified protocol is used in [5]. Here the verification of the integrity report and the property certificate is performed by a third party and the result alone is delivered to the attestation requester. Chen et. al [6] provide a protocol based on zero knowledge where the attestation requester can verify that a trusted platform satisfies a property without actually learning the hash values in the integrity report and the property certificate (thereby hiding the hash for better privacy). Sadeghi et. al [3], also propose a variation of the protocol where the TPM of the attesting platform itself securely verifies the IR and the property certificate, and generates an attestation certificate with the result which is then provided to the requester.

3 Authorization Logic for Property Attestation (ALOPA)

Logic programming has long been used to express access control policies as they provide mathematically well-founded semantics and sufficient flexibility to define policy rules. In logic, the authorization problem is expressed in terms of finding proof of a particular formula representing the permissions associated with resource access and a collection of suitable axioms representing policies. Credentials relevant to decision making become additional hypothesis to be used in the proof realized. Some popular authorization languages that are based on logic include RT [7], SD3 [8], BINDER [9], ASL [10], SDSI/SPKI [11] and TPL [12]. While the language itself is used to define

authorization policies, the decision is derived using an authorization engine. The engine accepts the policies, the request and the credentials supporting the request as inputs to arrive at the authorization decision.

Most authorization languages of today are designed for requestors that are human users or alike. This is obvious considering that human users place most requests for access to resources and hence the need to be controlled. The authorization policies are suited to better express conditions associated with the users and the authorization engine accepts the necessary user based credentials. For example, many popular authorization languages focus on features like role based access, delegation of rights, credential discovery, separation of duties, joint action policies and chinese wall policies while acceptable credentials include role and identity credentials, passwords or even biometric information. In the context of trusted computing platforms, authorizations can be at two different levels, namely the authorization of the user of the trusted computing platform and the authorization of the trusted computing platform itself. While legacy authorization languages are able to address the first requirement well, we believe that they are not suited for the second requirement, given the specific characteristics of trusted platforms. The languages can either be extended to address these specific requirements or may be used in conjunction with a separate policy language targeted for trusted platforms. In this vein, we have proposed a small language called ALOPA, which stands for Authorization Logic for Property Attestation in trusted platforms. ALOPA combines property based attestation and authorization of platforms. Access control policies specified using ALOPA govern the access of platforms to resources on the basis of the platform's identity and rules based on platform properties, which determine, for any platform and any resource, the types of accesses the platform is allowed on the resource. ALOPA can then be used in conjunction with any user based authorization system.

What is protected? The notion of a resource requiring protection in ALOPA is similar to that of any traditional authorization system. A resource can take different forms depending on the context in which the authorization is used. For example, in a peer to peer model, it is a file that is shared among peers, in a web service model, it is a web service, in a grid model, it is a piece of data requiring computation from grid nodes, or in a network access protection model, it is the network itself.

What to protect against? As discussed earlier, ALOPA protects resources not from unauthorized users but from unauthorized platforms. Each platform requiring access to a resource is to provide authentication and authorization credentials in order to gain access. A trusted platform is shipped with an Endorsement Key (EK) credential that is generated at the time of a platform manufacture. The private part of the key is stored inside the TPM and never leaves it. The TPM can then generate more private-public key pairs known as the Attestation Identity Keys (AIK). Using the EK credential, the platform is able to prove to a privacy CA that it is a genuine TPM, it stores the private AIK and gets the public AIK certified. We refer the reader to [1] for more details on the generation of EK and AIK key credentials. The public AIK credentials can then be used as authentication credentials for trusted platforms. Because a platform can generate any number of AIK credentials, producing a random AIK credential at the time of authorization can only prove that the requester is a genuine trusted platform. Alternatively, should the AIK credential be used

for the identification of the trusted platform also, then the platform should produce the same AIK credential for every request. Authorization credentials provide proof for properties satisfied by a platform. Authorization credentials are Integrity Reports and Property Certificates that are discussed in detail in Section 3. First, we provide some definitions.

Definition 3.1. Platform - *The term platform is short for a trusted computing platform. A platform is a collection of all hardware (including the Trusted Platform Module (TPM)) and all software elements installed on it.*

Definition 3.2. Component - *Each hardware and software element inside a platform is termed as a platform component. Each component performs a specific function, either individually or when combined with one or more components of the platform.*

Definition 3.3. Composition - *The term composition refers to the act of interconnecting one or more components of a platform in a certain fashion.*

The process of composition is an iterative one. First two components are connected. The resultant is considered as a new component and then another component is added to it. This procedure is repeated until the entire platform is built. With respect to security, it is easier to verify individual components' properties (which are smaller in size) rather than attempting to verify one large monolithic system. The goal of secure composition process is to ensure that a composed system preserves the security properties of the individual constituent components. This is in general difficult to achieve and designing securely composable properties is not a trivial task.

Definition 3.4. Property - *A property is defined as any attribute, characteristic or behaviour associated with a component or a platform. In this paper, a property is represented in the form pn_{pv} where pn is the name of the property and pv is the value of the property.*

In the context of security, we think of a property as a security service, mechanism or a function that is used to achieve a specific security goal. For example, if we consider data confidentiality as a security service and encryption as a security mechanism that can be used to provide the confidentiality service, then both data confidentiality and encryption can be thought of as security properties supported by a component. Alternatively, we may look at other definitions as defined in the Common Criteria specification for security functional requirements [13]. Common criteria classifies the security requirements into four levels namely Security Class, Security Family, Security Component and Element. A Security Class is on the top of the security hierarchy and includes a Security Family that addresses a particular security objective. A Security Family in turn consists of the Security Components (different from the components of a trusted platform). These Components are used to achieve the objectives of the Security Family. Every component further consists of Security Elements that enable the Components to realize the security objective. For example, user data protection is a Security Class, data authentication is a Security Family, basic data authentication is a Security Component and mechanisms like digital signatures are Security Elements of the Security Component. Essentially, each of these are security properties that are desirable from a software component. We now define the following policy expressions that are supported in ALOPA.

1. **Properties of platforms** - ALOPA supports access control rules based on the property requirements of a platform. Access to a resource is determined by the satisfaction of one or more properties by a platform.
2. **Properties of components** - ALOPA supports access control rules based on the property requirements of individual components of the platform. Access to a resource is determined by the satisfaction of one or more properties by one or more components of the trusted platform.
3. **Binary measurement values** - ALOPA supports access control rules based on the hash values of components. These rules are particularly useful to abstract pre-known hash measurements of a component to the properties that are satisfied.
4. **Properties to properties** - ALOPA supports the abstraction of one property to another related property. For example, Nagarajan et. al describe a pyramid model where low level properties can be abstracted to more meaningful high level properties [4]. Here, implementing an encryption algorithm (thereby satisfying encryption property) may be abstracted as providing confidentiality property. Such expressions are useful when many low level properties are combined to form a high level property (which is usually the case in normal implementations where many smaller functions or programs are combined to form larger programs).
5. **Targets for properties** - The language supports the expression of properties of a component bound to a specific target.

Definition 3.5. Target. *A target is a subject or an object to which a property of a component or a platform is bound to.*

For example let us consider an authentication system where all users are authenticated using passwords and all log in attempts are recorded. Here, the authentication system (component) supports authentication (property) of all users (targets). The system (component) securely logs (property) all login outcomes (target).

6. **Prerequisite conditions** - ALOPA supports the expression of prerequisite conditions for a component or a platform to satisfy a given property.

Definition 3.6. Prerequisite. *A given property of a component or a platform is satisfied only if all the prerequisite properties have been satisfied.*

For example, for secure audit data generation (property), a reliable time stamp generator is a prerequisite.

7. **Composition of properties** - The language supports the expression of composition [Definition 3.3] of components and properties in a platform. The outcome of a composition is different depending on the components and the properties that are composed. For example, if we imagine that there exists two components a and b that satisfy two different properties pa and pb respectively. When a and b are composed together, a new component c is generated that does not satisfy pa and pb but satisfies pc. Such conditions are useful to express authorization policies. Without such conditions, one can interpret that c also satisfies pa and pb. Determining the outcome of a composition is beyond the scope of this paper. Some literature on composition can be found here [14], [15]. In ALOPA, we assume that compositions and their outcome are pre-known to the policy administrators. (This is a reasonable assumption because policy administrators are not expected to perform such tasks)

and ALOPA is used only to express such conditions. The following preliminaries are defined in ALOPA.

1. **Constant Symbols:** Every member of $PF \cup C \cup P \cup T \cup O \cup A \cup PER \cup \mathbb{N}$ is a constant. Here, PF is a set of platforms, C is a set of components, P is a set of properties, T is a set of targets, O is a set of objects, A is a set of actions, PER is a set of permissions and \mathbb{N} is a set of natural numbers. Each member of the set is represented in the form set_i where i is a member of the set \mathbb{N} . For example, pf_1, pf_2, \dots, pf_n represent the members of the set PF.
2. **Variable Symbols:** We define seven sets of variable symbols $V_{pf}, V_c, V_p, V_t, V_o, V_a, V_{per}$ ranging over the sets PF, C, P, T, O, A, PER respectively.
3. **Predicate Symbols:** The following predicate symbols are defined in ALOPA. Each n-ary predicate symbol is represented as $p(t_1, t_2, \dots, t_n)$ where p is the name of the predicate and n is its arity.
 - (a) **HasC** is a binary predicate symbol. Both the arguments of HasC are members of C. The predicate defines a hierarchal relationship between two components in a given platform. For example, $HasC(c_1, c_2)$ is read as component 'c₁' has (or contains) the component 'c₂'.
 - (b) **HasPF** is a binary predicate symbol. The first argument is a member of PF and the second argument is a member of set C. It defines a hierarchal relationship between a platform and a component. For example, $HasPF(pf_1, c_1)$ is read as a platform pf_1 has the component c_1 .
 - (c) **IFlow** is a ternary predicate symbol. The first two arguments are members of C and the third argument is either 'read' or 'write'. The predicate defines a relationship between two components based on the type of information flow between them. For example, $IFlow(c_1, c_2, write)$ is read as component c_1 writes to c_2 .
 - (d) **SatC** is a ternary predicate symbol. The first argument is a member of C, the second argument is a member of P and the third argument is a member of T. It defines the relationship between a component and the property it satisfies for a given target. For example, $SatC(c_1, p_1, t_1)$ is read as component c_1 satisfies a property p_1 for the target t_1 . Target field is optional.
 - (e) **SatPF** is a ternary predicate symbol. The first argument is a member of PF, the second argument is a member of P and the third argument is a member of T. It defines the relationship between a platform and the property it satisfies for a given target. For example, $SatPF(pf_1, p_1, t_1)$ is read as platform pf_1 satisfies a property p_1 for the target t_1 . Target field is optional.
 - (f) **PreReq** is a sestary predicate symbol. PreReq defines the 'prerequisite' relationship between two SatC predicates. For example, $PreReq((c_1, p_1, t_1), (c_2, p_2, t_2))$ is equivalent to $\neg SatC(c_1, p_1, t_1) \leftarrow \neg SatC(c_2, p_2, t_2)$.
 - (g) **Do** is a quaternary predicate symbol. The first argument is a member of PF, second argument is a member of O, third argument is a member of A and the fourth argument is a member of the set Permissions = {allow, deny}. It defines the authorizations that are held for each platform on each object. For example, $Do(pf_1, o_1, a_1, allow)$ is read as platform pf_1 is allowed to perform action a_1 on the object o_1 .

Every n-ary predicate symbol of the form $p(t_1, t_2, \dots, t_n)$ is termed as an atom. The term literal is used to denote an atom or its negation. For example, $HasC(c_1, c_2)$ and $\neg HasC(c_1, c_2)$ are both examples of literals. We use Horn-Clause logics [16] to represent our rules in ALOPA. All rules are restricted to the form $head \leftarrow body$, where \leftarrow is a right-to-left implication symbol, $head$ is a literal, $body$ is a conjunction of literals and the head of each rule is the consequent of the body. If the head is empty, then the rule is a fact.

Definition 3.7. Component-Property (CP) Rule- A CP rule is a rule of the form:

$$SatC(c, p, t) \leftarrow L_1 \wedge L_2 \wedge \dots L_n.$$

where c, p, t are elements of the sets C, P and T respectively, $n \geq 0$, $L_1 \dots L_n$ are either SatC or HasC literals (a) A rule with a SatC literal in the body indicates the abstraction of one property to another. The component in the head and the body must be the same. Examples (a) and (b) illustrate this. (b) The rule may also be used to show that a component may satisfy one property if one of its sub-component satisfies the same (or a different) property. This is shown in example (c) below. Note that the rule may contain any number of literals.

$$(a) SatC(c_1, p_1, t) \leftarrow \forall t \in T : SatC(c_1, p_2, t)$$

$$(b) SatC(Browser, Trusted_True) \leftarrow$$

$$SatC(Browser, Hash_e2c182bbb85c2e3a5fcae1936c5900cf91dd7743)$$

$$(c) SatC(c_1, p_1, t) \leftarrow \forall t \in T : SatC(c_2, p_2, t) \wedge HasC(c_1, c_2, t)$$

First rule reads as: for all possible targets, for a component c_1 to satisfy a property p_1 , c_1 must satisfy a property p_2 . For example, an application (c_1) that can perform an AES algorithm (p_2) can perform encryption (p_1), a browser (c_1) that can perform SSL (p_2) is expected to be secure (p_1), a Web service (c_1) that can perform identify based authentication (p_2) can perform authentication (p_1). This rule is particularly useful for abstracting properties at one level to properties at a different level as already discussed (security mechanism based property to security service based property etc). The second rule transforms the 160-bit binary measurement of a browser component to a property (trusted) of the component. The third rule reads as: for all possible targets, for a component c_1 to satisfy a property p_1 , c_2 must satisfy p_2 and c_1 must contain c_2 . For example, an application (c_1) can perform auditing operations (p_1) of all the inputs and outputs (t) if it (c_1) contains another sub-component (c_2) that records all audit logs (p_2).

Definition 3.8. Platform-Property (PP) Rule 2- A PP rule is a rule of the form:

$$SatPF(pf, p, t) \leftarrow L_1 \wedge L_2 \wedge \dots L_n.$$

where pf, p, t are elements of the sets PF, P and T respectively, $n \geq 0$, $L_1 \dots L_n$ are either SatC, SatPF or HasPF literals. (a) A rule with a SatPF literal in the body indicates the abstraction of one property to another. The platform in the head and the body must be the same. Example (a) illustrates this. (b) The rule may also be used to show that a platform may satisfy one property if one (or more) of its component satisfies the same (or a different) property. This is shown in example (b) below. Note that the rule may contain any number of literals (as in example (c)).

- (a) $SatPF(pf_1, p_1, t) \leftarrow \forall t \in T : SatPF(pf_1, p_2, t)$
 (b) $SatPF(pf_1, p_1, t) \leftarrow \forall t \in T : SatC(c_2, p_2, t) \wedge HasPF(pf_1, c_2)$
 (c) $SatPF(pf_1, p_1, t) \leftarrow \forall t \in T :$
 $SatC(c_2, p_2, t) \wedge HasPF(pf_1, c_2) \wedge (SatPF(pf_1, p_3, t))$

Rule (a) reads: A platform pf_1 satisfies the property p_1 if the platform also satisfies property p_2 . For example, in the context of network admission control, a platform (pf_1) is considered known (p_1) if it has an IP address x (p_2). Rule (b) reads: a platform can have a property p_1 if it has a component c_2 that satisfies another property p_2 . For example, a platform is considered to be safe (p_1) if it has an antivirus software (c_2) which is up-to-date (p_2). Rule (c) adds an extra condition to Rule (b) that the platform must also satisfy property p_3 . For example, a platform is considered to be safe (p_1) if it has an antivirus software (c_2) that is up-to-date (p_2) and if the platform is known (p_3).

Definition 3.9. Access Control Rule- An access control rule is a rule of the form:

$$Do(pf, o, a, per) \leftarrow L_1 \wedge L_2 \wedge \dots L_n.$$

where $L_1 \dots L_n$ are SatPF literals. An authorization rule is used to specify the authorization permissions for a platform based on the properties satisfied by the platform.

- (a) $Do(pf_1, a_1, o_1, per_1) \leftarrow \forall t \in T : SatPF(pf_1, p_1) \vee SatPF(pf_1, p_2)$

Rule (a) reads: a platform pf_1 has the permission per_1 to perform an action a_1 on an object o_1 if the platform either satisfies property p_1 or property p_2 . For example, a platform (pf_1) is allowed (per_1) to connect (a_1) to a network (o_1) if it has the IP address $x.x.x.x$ (p_1). Table 1 provides a summary of the literals allowed for CP, PP and Do rules.

Definition 3.10. Authorization Specification

An authorization specification is a collection of authorization rules that have been defined in Section 3. The authorization specification determines whether an action request is allowed or denied for a trusted platform. Formally, An authorization specification language for property attestation is a set of CP, PP and Access Control rules.

Table 1. Literals allowed for CP,PP and Do rules

Literals in CP rules	
Sat literals	$SatC$
Has literals	$HasC$
Literals in PP rules	
Sat literals	$SatC, SatPF$
Has literals	$HasPF$
Literals in Do rules	
Only Sat literals	$SatPF$

4 Authorization Derivation

An authorization logic program in ALOPA comprises of a set P of program clauses. The set P consists of a set of rules and set of facts. The program together with a query Q describes the deduction. For example, if Program Clause P , Facts: e.g. a, b , Rules: e.g. $c \leftarrow a \wedge b$, Query Q and Goals: e.g. c . The subsequent behaviour of the logic program is based on establishing the deduction using rule resolution [17], i.e by showing that the set of clauses $\{a, b, c \leftarrow a \wedge b, \neg c\}$ is unsatisfiable. Facts in ALOPA are PreReq predicates as defined in the section above. The rules in ALOPA program are either CP, PP or access control rules as defined in the section above. A query is a request for permission to access a resource that is protected by the authorizer. A query is usually accompanied by the authorization credentials. A rule resolution determines if the query is satisfiable (and hence the goal achievable) given the set of facts and rules.

In this section, we discuss about the authorization credentials and rule resolution. A query is first made by a trusted platform, also called the Attesting Party (AP). The query is received by an authorizer, also called the Attestation Requester (AR). AR protects resources for which AP requests access. AR administers ALOPA policies and runs an ALOPA authorization engine. When AR receives a query from AP, it forwards the query along with AP's credentials to AR's authorization engine. The authorization engine uses resolution rules to determine if the query is satisfiable. Once the evaluation is complete, it forwards its decision to AR based on which access is either allowed or denied.

Authorization credentials: Authorization credentials determine if a platform has the necessary attributes for a request to be serviced. Property certificates defined earlier provide evidence for properties that are satisfied by a platform or a component. A property certificate is issued by a trusted third party who believes that a property is available in a platform/component. Property certificates are issued either for the entire platform or are issued for individual components. In the case of a platform, a property certificate contains information about the identity of the platform and the properties that are satisfied by the platform. The property certificate issued for a component includes information about the identity of the component, the properties satisfied by that component and a list of sub-components of the certified component. This is because, AR can not only be interested in the properties of the main component but also in the properties of the sub-component. Then, the property certificate can be used to determine the HasC relationships of the components. For example, let AR define a policy that AP's kernel and the kernel's sub-component - kernel configuration file should be integrity protected. Then AP should present AR with a property certificate of the kernel (that determines if the kernel is integrity protected and what its configuration file is) and the property certificate of that configuration file. (It should be noted that the sub-component should present its own property certificate that proves its integrity.) An Integrity Report is also an authorization credential in the context of trusted platforms as they provide evidence for the hash value attributes of the components. Now, given the integrity report and the property certificates, the authorizer must to derive the following information.

1. **HasPF:** The Authorizer derives the HasPF information from the *Integrity Report*. Please recall that IR is a report that is generated at the time of attestation with the list of components measured along with their measurements. Using this list, the

authorizer derives the HasPF literals and stores them as derived facts in a derived-facts base. This information will then be used by Algorithm 1 for resolution.

2. **HasC:** The Authorizer derives the HasC information using the *Property Certificates* of a component. HasC defines what sub-components are present in a main component. Using this list, the authorizer derives the HasC literals and adds them to the derived-facts base. This information will be used by Algorithm 2 for resolution.
3. **SatPF and SatC:** The authorizer derives the SatPF and SatC information from the property certificates of the platform and components respectively. It adds this information to the derived-facts base. This will then be used in algorithms 1 and 2 for resolution.

The following are the steps of resolution in the policy engine. The policy engine begins the resolution from the Do rule using algorithm 1. It calls the algorithm 2 if necessary. Any standard resolution algorithm implementation for logic programs like Selective Linear Definite (SLD) [18] resolution with back tracking may be used to implement these algorithms.

Algorithm 1. The policy engine starts with the Do rule and reads each literal in the body of the Do rule.

- For each SatPF literal in the Do rule
 - It first checks if the SatPF literal is available as a derived fact. It also verifies that any pre-requisites for SatPF are satisfied before marking the SatPF literal as true.
 - If the SatPF literal is not found in the derived-facts base, it checks if there are PP rules with SatPF as head of the rule. It tries to resolve the PP rule to determine if SatPF is true. This process is recursive.
- For each HasPF literal in the Do rule
 - It checks if HasPF is in the derived-fact base. If yes, it marks HasPF as true.
- For each SatC literal in the Do rule or PP rule from step above
 - It calls the CP resolution algorithm
- If all literals in Do are true, the algorithm outputs the permissions in Do.

Algorithm 2. The policy engine starts with the CP rule and reads each literal in the body of the CP rule.

- For each SatC literal in the CP rule
 - It first checks if the SatC literal is in derived-facts base. If so, it also verifies that any PreReqs for SatC are satisfied before marking SatC literal true.
 - If the SatC literal is not in derived-facts base, it checks if there are CP rules with SatC as head of the rule. It tries to resolve the CP rule to determine if SatC can be satisfied. This process is recursive.
- For each HasC literal in the CP rule
 - It checks if HasC is in derived-facts base. If yes, it marks the HasC literal true.

5 Application Scenario

In this section, we demonstrate the application of ALOPA using an example in the context of network access control. Network Access Control (NAC) is a suite of software that tries to ensure that every endpoint client that requests access to the network satisfies a set of pre-determined security policies. Popular network access control systems include Microsoft's Network Access Protection (M-NAP) [19] and Cisco's Network Access Control (C-NAC)[20]. In the same vein, the Trusted Computing Group supports the Trusted Network Connect (TNC) [21] initiative to build an open, non-proprietary standard that enables the enforcement of security requirements for endpoints connecting to the corporate network. Currently, the use of TPM is optional and decision making is based only on the integrity report validation. ALOPA is useful to extend such functionality with property based attestation.

A NAC system includes one or many Security Health Validators (SHVs) that perform health assessment checks on the requesting clients. For example, M-NAP comes with Windows Security Health Validator (WSHV) that checks for the desirable properties in a client system. Today, there are also many third party vendors who provide their own SHV plug-in modules for M-NAP. For example, Blue Ridge Networks provide an Intrusion Detection SHV [22], popular anti-virus vendors also provide SHVs like McAfee SHV, Symantec SHV for enhancing network access control and UNET systems' UNET SHV [23] for providing more fine granular network access protection. The Network Policy Server (NPS) can also define other policies outside the scope of the SHVs. For example, it can check the IP address of the client and check if the client supports Microsoft's encrypted authentication using either MS-CHAP v2 [24]. Table 2 summarizes these properties in column (a). Columns (b) and (c) provide the corresponding component name/ platform id and property names (as used in ALOPA). For every SHV in NAP, a corresponding Security Health Agent (SHA) component is installed in the client. The health agents reside in the client system, collect and report information that the SHV requests for. Using this information, the NPS policy engine makes network access control decisions.

In such a system, one can think of at least two possible concerns for the client as it requests connection to the network.

(a) The client may be unwilling to install a third party health agent in its system. If a client system makes frequent requests to connect to the network, the health agent usually resides in the client permanently. This may also not be desirable for the client.

(b) Even if the client allows the agent to reside in its system, how can it be sure that the health agents perform only legitimate tasks and not cause any harm to the system.

Property Based Attestation can be used to address both these concerns. For (a), health agents may not be installed on the client system. Rather, if the client is a trusted platform, then then it can use property based attestation to prove to the NPS that it satisfies all the properties requested by the NPS. As discussed earlier, the client can obtain the necessary property certificates from a certification authority. The certification authority can vouch that the required properties are available in the client. The policy engine of the NPS can then use these property certificates to resolve its access control policies. A few examples of such policies in ALOPA is illustrated here.

Table 2. Desirable properties of a client system

Windows Security Health Validator (WSHV)		
(a)	(b)	(c)
Property Requirement	Component name/Platform ID	Property name
If a firewall is enabled	Firewall	Enabled
If an antivirus is enabled	Antivirus	Enabled
If the antivirus is up-to-date	Antivirus	Uptodate
If anti-spyware is enabled	Anti-Spyware	Enabled
Check if Windows update in enabled	Windows	Uptodate
UNET SHV		
If a file exists in the system	File	Exists
The file size	File	Size
The file binary hash measurement	File	Hash
If a program is installed in the system	Program	Installed
The version of the Windows OS	Windows	Version
Network Policy Server		
The identity of the platform	Platform ID (AIK)	Identity
The IP address of the platform	Platform ID (AIK)	IP
If MSCHAP v2 is supported	Platform ID (AIK)	EncryptedAuthentication

1. $Do(AIK_1, Network, Connect, Allow) \leftarrow SatPF(AIK_1, Trusted)$
2. $SatPF(AIK_1, Trusted_True) \leftarrow$
 $SatPF(AIK_1, Known_True) \wedge SatPF(AIK_1, Secure_True)$
3. $Sat(AIK_1, Known_True) \leftarrow SatPF(AIK_1, IP_192.70.4.6)$
4. $SatPF(AIK_1, EncryptedAuthentication_True) \leftarrow$
 $SatC(Authenticator, MSCHAPv2_True) \wedge HasPF(AIK_1, Authenticator)$
5. $SatPF(AIK_1, Secure_True) \leftarrow SatPF(Firewall, Enabled_True) \wedge$
 $SatPF(Antivirus, Enabled_True) \wedge SatPF(Antivirus, Uptodate_True) \wedge$
 $SatPF(Spyware, Enabled_True) \wedge SatPF(Windows, Uptodate_True) \wedge$
 $SatPF(AIK_1, EncryptedAuthentication_True)$

Here, the platform identified by the AIK_1 key is allowed to connect to the network if it is trusted. In order for it to be trusted, it must be a known system and must be secure. A system is known if it has the IP 192.70.4.6. A system is secure if it has firewall enabled, if the antivirus is enabled and up to date, if the anti-spyware is enabled, if Windows is up to date and if the platform supports encrypted authentication. A platform can support encrypted authenticated if it supports the MSCHAPv2 algorithm. If the client can provide property certificates and integrity report from which the NPS policy engine can resolve these policies, then the NPS can be certain that the properties are available in the client in order to allow access to the requested service.

To address issue (b), both the client and the NPS from where the SHA is installed should support property based attestation. The health agent can obtain a property certificate from a certification authority indicating the properties that it satisfies. This can include properties like absence of malware, the different program calls that the agent makes in the system, to which servers the agent is allowed to communicate or even binary measurements. If the client is satisfied with the status of the agent component, it

allows the agent to be installed. An example of a client policy is given below. Here, the agent must either produce an integrity report with the required hash or it must produce a property certificate that proves that its is free from malware.

$$\text{SatC}(\text{Agent}, \text{Safe_True}) \leftarrow \text{SatC}(\text{Agent}, \text{Malware_False})$$
$$\text{SatC}(\text{Agent}, \text{Safe_True}) \leftarrow$$
$$\text{SatC}(\text{Agent}, \text{Hash_5ce2e6f40299204d94dfc0abf19fa8ab52d6c211})$$

6 Conclusion

In this paper, we have discussed how property based attestation can be used to enhance authorization in trusted platforms at the time of access request to resources. We have proposed a logic language ALOPA for describing and evaluating authorizations for property based attestation in trusted platforms. ALOPA is used to define access control policies for trusted platforms such that only platforms with the required properties are allowed access to the requested resource. Finally, we have demonstrated the use of ALOPA with a small practical example involving network access control.

As future work, we would like to extend ALOPA in detail to include the IFlow predicates. We are also working on other aspects of the language like conflict resolution. Then, we would like to investigate if specific conditions in user based authorization such as Chinese wall policies, delegation, joint action policies are applicable in the context of property attestation as well. Finally, we would also like to combine ALOPA based platform authorization policies with traditional user authorization policies for better security decision making.

References

1. Trusted Computing Group: TPM Main - Part 1 Design Principles, Version 1.2, Revision 103 (July 2007)
2. Poritz, J., Schunter, M., Herreweghen, E.V., Waidner, M.: Property Attestation—Scalable and Privacy-Friendly Security Assessment of Peer Computers. Technical report, IBM Research (May 2004)
3. Sadeghi, A.R., Stübke, C.: Property-Based Attestation for Computing Platforms: Caring about Properties, not Mechanisms. In: NSPW 2004: Proceedings of the 2004 Workshop on New Security Paradigms, pp. 67–77. ACM, New York (2004)
4. Nagarajan, A., Varadharajan, V., Hitchens, M.: Trust Management for Trusted Computing Platforms in Web Services. In: STC 2007: Proceedings of the 2007 ACM Workshop on Scalable Trusted Computing, pp. 58–62. ACM, New York (2007)
5. Nagarajan, A., Varadharajan, V., Hitchens, M., Arora, S.: On the Applicability of Trusted Computing in Distributed Authorization Using Web Services. In: Atluri, V. (ed.) DAS 2008. LNCS, vol. 5094, pp. 222–237. Springer, Heidelberg (2008)
6. Chen, L., Landfermann, R., Löhr, H., Rohe, M., Sadeghi, A.R., Stübke, C.: A Protocol for Property-Based Attestation. In: STC 2006: Proceedings of the first ACM workshop on Scalable Trusted Computing, pp. 7–16. ACM, New York (2006)
7. Li, N., Mitchell, J.C., Winsborough, W.H.: Design of a Role-Based Trust Management Framework. In: Proc. IEEE Symposium on Security and Privacy, Oakland (May 2002)

8. Jim, T.: SD3: A Trust Management System with Certified Evaluation. In: SP 2001: Proceedings of the 2001 IEEE Symposium on Security and Privacy, Washington, DC, USA, p. 106. IEEE Computer Society, Los Alamitos (2001)
9. DeTreville, J.: Binder - A Logic-Based Security Language. In: SP 2002: Proceedings of the 2002 IEEE Symposium on Security and Privacy, Washington, DC, USA, p. 105. IEEE Computer Society, Los Alamitos (2002)
10. Jajodia, S., Samarati, P., Subrahmanian, V.S.: A Logical Language for Expressing Authorizations. In: SP 1997: Proceedings of the 1997 IEEE Symposium on Security and Privacy, Washington, DC, USA, p. 31. IEEE Computer Society, Los Alamitos (1997)
11. Rivest, R.L., Lampson, B.: SDSI - A Simple Distributed Security Infrastructure. Presented at CRYPTO 1996 Rumpsession (1996)
12. Herzberg, A., Mass, Y., Michaeli, J., Ravid, Y., Naor, D.: Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers. In: SP 2000: Proceedings of the IEEE Symposium on Security and Privacy, Washington DC, USA, p. 2. IEEE Computer Society, Los Alamitos (2000)
13. Common Criteria Sponsoring Organizations: Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Components, Version 3.1 Rev 1-Nat'l Inst. of Standards and Technology CCMB-2006-09-002 (September 2006)
14. Roscoe, A.W., Wulf, L.: Composing and Decomposing Systems under Security Properties. In: CSFW 1995: Proceedings of the 8th IEEE workshop on Computer Security Foundations, Washington, DC, USA, p. 9. IEEE Computer Society, Los Alamitos (1995)
15. Zakinthinos, A.: On the Composition of Security Properties. PhD thesis, University of Toronto (1996)
16. Horn, A.: On Sentences which are True of Direct Unions of Algebras. *J. Symb. Log.* 16(1), 14–21 (1951)
17. Gallier, J.H.: *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Harper & Row Publishers, Inc., New York (1985)
18. Kowalski, R.: Predicate Logic as Programming Language. In: IFIP Congress, pp. 569–574 (1974)
19. Microsoft Corporation: Network Access Protection Platform Architecture (February 2008)
20. Cisco Systems: Network Admission Control Documentation Reference Guide. 2.0 edn. (April 2006)
21. Trusted Computing Group: Trusted Network Connect (TNC) Architecture for Interoperability Version 1.3 (April 2008)
22. Iverson, E.: NAP Enhanced to Secure Endpoints on and off the Enterprise. Blue Ridge Networks (2008)
23. Seongyon, H., Eunseok, C., Wonseok, C., Jihyun, L., Youngman, P.: UNETSHA - Plug-in for Extending Microsoft NAP. UNET System Inc., Korea (2008)
24. Zorn, G.: Microsoft PPP CHAP Extensions, Version 2. RFC 2759 (2000)

Employed BPN to Multi-sensors Data Fusion for Environment Monitoring Services^{*}

Wen-Tsai Sung

Department of Electrical Engineering, National Chin-Yi University of Technology
35, 215 Lane, Sec. 1, Chung Shan Road, Taiping, Taichung, Taiwan
songchen@ncut.edu.tw

Abstract. The real-time system uses a back-propagation network (BPN) with associative memory for recognition and classification in multi-sensors data fusion. This study attempts to apply classification fusion technology to the real-time signals recognition of multi-sensors data in a wireless sensor networks (WSNs) system with a node–sink mobile network structure. These wireless sensor network systems include temperature, humidity, ultraviolet, and illumination four variable measurements for environment monitoring services (EMS). Remote engineers can manage the multi-sensors data fusion using the browser, and the WSNs system then classification the data fusion database via the Internet and mobile network. Moreover, the data fields of each sensor node contain the properties and specifications of that pattern, except in the case of engineering components. The database system approach significantly improves classification data fusion system capacity. The classification fusion system examined here employs parallel computing, which increases system data fusion rate. The classification fusion system used in this work is an Internet based node–sink mobile network structure. The final phase of the classification fusion system applies database BPN technology to processing data fusion, and can solve the problem of spurious states. The system considered here is implemented on the Yang-Fen Automation Electrical Engineering Company as a case study. The experiment is continued for 4 weeks, and engineers are also used to operating the web-based classification fusion system. Therefore, the cooperative plan described above is analyzed and discussed here. Finally, these papers propose the tradition methods compare with the innovative methods.

Keywords: back-propagation network, wireless sensor networks, environment monitoring services, data fusion.

1 Introduction

Wireless sensor networks have emerged as a new information-gathering paradigm based on the collaborative effort of a large number of sensing nodes. This paper describes the application of BPN technology in the problem domain of sensor data fusion. A sensor network consists of many spatially distributed sensors called nodes,

^{*} This work was supported in part by the National Science Council Taiwan, through it's grand no. NSC- 97-2218-E-167-001.

these nodes are used to monitor or detect various kinds of changes in vibration, pressure, movement or pollutant levels. In this investigation, we design four various sensors in the experiment circuit boards include temperature, humidity, illumination, and ultraviolet measurements for EMS. These nodes are usually small and inexpensive in order to allow them to be deployed on a large scale. These are sensors usually have a wireless link which can be used to extract the information captured by the sensor.

A sensor node has a small microcontroller, and an energy source, usually a battery. In order to meet the objective of these sensors being small and low-cost, resources in terms of energy, memory, computational speed and bandwidth are severely constrained. The sensors use each other to transport data to a monitoring entity. Because each sensor has a limited energy supply, sensors must conserve their energy if the network is to last a long time. Wireless, database technology such as queries, and networking technology especially, multi-hop routing protocols to communicate with other nodes is crucial technologies. Wireless technology is used in a type of network, a wireless sensor network. ZigBee is a wireless protocol used by IEEE 802.15.4 association and ZigBee alliance. In this study, the networks need be able to self-organize via ZigBee wireless protocol. The same type of aggregate data with variances needs to be grouped and fused into a single datum for intelligent interpretation. Major limitations included limited storage and power. A special node which connects to a computer and outside the network is called the gateway node.

This paper describes the application of BPN technology in the recognition and classification of multi-sensors data fusion. The first section of this paper introduces and reviews the problem presented by sensor fusion. The second section provides the background on BPN and sensor data fusion. The subsequent section discusses the domains where neural-network is applied for sensor data fusion varying as wide as intelligent waste-water management to military surveillance. We provide a model for wide-area surveillance using BPN based multi-sensors data fusion. Finally, we also discuss how precise modifications to BPN can improve the classification level of sensor fusion.

2 Related Literatures

WSNs are composed of a large number of nodes with sensing capability [1]. The applicability of such networks includes several areas such as environmental, medical, industrial, and military applications. Usually, WSNs have strong constraints regarding energy resources and computational capacity. In addition, these networks demand self-organizing features to autonomously adapt themselves to eventual changes resulting from external interventions, reaction to a detected event, or requests performed by an external entity.[2]In general, WSNs are deployed in environments where sensors can be exposed to conditions that might interfere with the sensor readings or even destroy the sensor nodes. As a result, sensor measurements may be more imprecise than expected, and the sensing coverage may be reduced. A natural solution to overcome failures and imprecise measurements is to use redundant nodes that cooperate with each other to monitor the environment. However, multi-sensors data fusion comprises theories, algorithms, and tools used to process several sources of information generating an output that is, in some sense, better than the individual sources. [3].

This investigation implements a BPN in multi-sensors data fusion for recognition and classification. The type of recurrent neural network used is known as the multi-layer feed-forward network. The BPN provides the basis for nonlinear associative memory. Significantly, the BPN is very effective in data fusion for recognition and classification. [4] [5] Supervised learning is achieved through the error back-propagation algorithm. In the literature [6], the authors described multi-source data fusion and management for virtual wind Tunnels and physical wind tunnels, the system always adopts the latest data fusion and database conceptions via BPN. In Loskiewicz et al. researches [7], a diagnostic system design which performs evidence aggregation from many sensors in order to automate the interpretation of vibration spectra. The decision system proposed is an active system which its module is a BPN system. The method proposed is very general and can be used for any problem involving aggregation of decisions for the purpose of classification. According to above-mentioned, the data fusion employed BPN is better than traditions methods in recognition and classification based on WSNs. [8] [9] [10] [11] [12] [13][14].

3 System Architecture for EMS

WSNs combining the mobile computing, telecommunication and sensing equipments can operate automatically with least power consumption. The WSNs applications include the monitoring of wild animals, environment monitoring/forecast and health monitoring. This paper intends to use Motes wireless sensor networks, which enable data collection covering soil and air humidity, air temperature, light and ultraviolet. All data from every sensor can be transmitted via ZigBee network transmission protocol, thus forming mobile WSNs. The EMS data are sent back to the rear database via wireless network and mobile telecom network, thereby contributing to real-time and continuous monitoring of drought. The EMS model decision system automatically analyses the environmental drought data and the results are sent to the end users in real-time. This reduces human error for a more accurate EMS. In fig.1, sensor node is capable of detecting and collecting the environmental data. Sensor node processes the

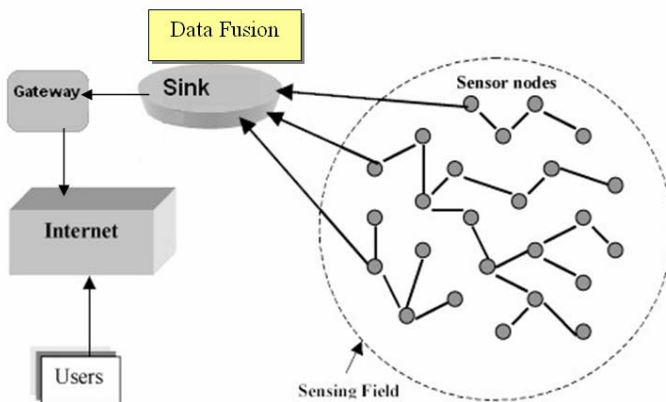


Fig. 1. The system architecture for EMS based on WSNs

collected data and transmits them to the sink node. Sink node is a gateway node, which is responsible for receiving the sensor data and re-transmitting these data to the manager node via Internet or mobile networks. Manager node is responsible for processing and displaying the sensor data.

Figure 2 depicted our WSN nodes device that comprises the sensing units, processing units, transceiver units and power units. The functionality of the units is described as follows.

- (1) Sensing Unit. A sensing unit comprises the sensor and the analog-to-digital converter. The sensor is responsible for detecting and collecting the environmental data, which represent with the analog signals. The analog-to-digital converter converts the analog signals into the digital data and sends the data to the processing unit. Sensing Unit.: include temperature, humidity, ultraviolet, and illumination.
- (2) Processing Unit. Processing unit comprises the processor and the storage unit. Storage unit stores the collected environmental data. Processor processes the data according to the pre-defined program codes.
- (3) Transceiver Unit. Transceiver unit is responsible for the communications between the sensor devices.
- (4) Power Unit. Power unit provides the electric power and is the most important unit of a WSN device.

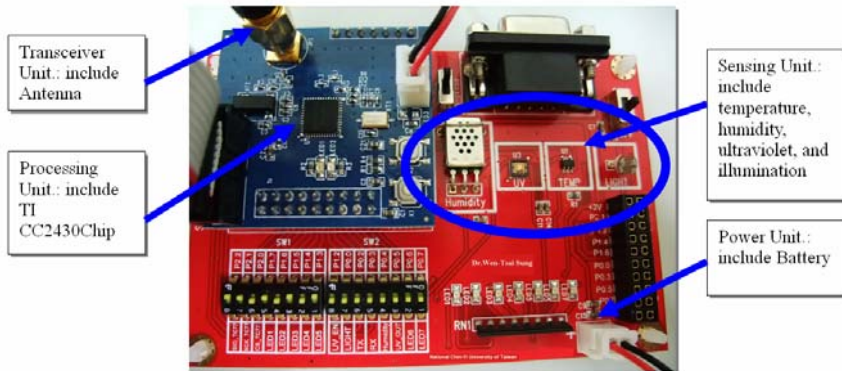


Fig. 2. These devices of sensor node in EMS are built via our researches

4 BPN Algorithm for Data Fusion in Recognition and Classification

The acquired data is first subjected to preprocessing step in a sensor node with various sensor components. Besides filtering for noise removal, this step also processes the signal for achieving invariance to selected inspection parameters. For instance, in the case of inspection data fusion at different inspection frequencies the signals are first transformed to an equivalent signal at a reference value of the inspection frequency parameter. Similarly, the overall classification performance of the system can be rendered invariant to other selected parameters [15]. In the second step,

discriminatory features in the signal are extracted. Feature extraction serves to reduce the length of the data vector by eliminating redundancy in the signal and compressing the relevant information into a feature vector of significantly lower dimension. The Discrete Wavelet Transform (DWT) is particularly effective at extracting features at multiple resolution levels in ultrasonic signals which are inherently non-stationary in nature [16]. A second set of features based on Principal Component Analysis (PCA) also calculates the statistical properties of a set of neighboring A-scans [17].

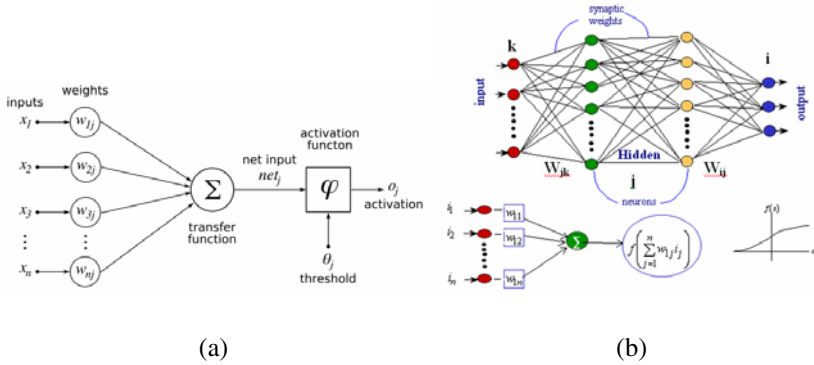


Fig. 3. Neural network general neuron and classification model

Neural networks are perhaps the most commonly used algorithm in automated classification of signals [17]. These networks have proved to be extremely effective in learning subtle differences in signals from various classes (indications) as shown in Fig. 3. A neural network classifier with the error back-propagation training algorithm is used in the signal classification system developed in this study. The objectives of the paper were focused on demonstrating the four different signals classification system on TI CC2430 Chip. The demonstrations were conducted using BPN procedure qualification test but on a smaller scale. The final phase of this work will focus on the development of a commercial quality windows-based software package for automated EMS data fusion. This paper utilizes the most prevailing BPN algorithm to analyze the potential degrees for EMS measurements. The BPN algorithm is a typical supervised learning network [18], which is to learn the internal reflection and regulations between inputs and outputs. The regulations are the synaptic weights of network neurons. For analyzing any new cases, the input values or independent variables are inputted into the neural network and get the inferential related output values quickly. BPN have three system layers and described as follows:

- (1) Input Layer comprises the inputs of the BPN and represents the initial values of decision.
- (2) Hidden Layer comprises the neurons, which are responsible for adjusting the synaptic weights of neuron linkages and determining the suitable synaptic weights. To have accuracy results, the hidden layer is composed of several sub-layers to learn the internal reflection and regulations between inputs and outputs.
- (3) Output Layer comprises the outputs of the BPN and represents the final decision results at this training operation.

The control procedure of the EMS based on BPN algorithm divides into the following operation steps:

- (1) Set up the network parameters.
- (2) Set up weighted matrixes, i.e., W_{xh} and W_{hy} , and the initial values of bias vector, i.e., θ_h and θ_y , by uniformly random numbers.
- (3) Calculate the output quantity of the hidden layer.
- (4) Set up the tolerant difference quantity between the output layer and the hidden layer.
- (5) Calculate the difference quantity, i.e., δ , between the output layer and the hidden layer.
- (6) Determine whether the difference quantity between the output layer and the hidden layer is greater than the tolerant difference quantity Φ . If the difference quantity δ is smaller than the tolerant difference quantity Φ , the optimal regression model is obtained.
- (7) If the difference quantity δ is greater than the tolerant difference quantity Φ , the weighted matrixes W_{xh} and W_{hy} , and the corrections of bias values θ_h and θ_y in the output Layer and the hidden layer have to be computed.
- (8) Revise the weighted matrixes and the bias values in the output layer and the hidden layer, and repeat steps (3) to (7) until the difference quantity lies within the range of the tolerant difference quantity.
- (9) Finally, compare the correlation of sensitivity correction to find out the optimal regression model.

5 Error Back-Propagation Algorithm for WSNs

We consider a multilayer perceptron with three layers (input, hidden, output) as shown the figure 3. The connection weights between the j^{th} neuron in the hidden layer and the k^{th} neuron in the input layer is denoted by w_{jk} . Similarly the connection weights between the i^{th} neuron in the output layer and the j^{th} neuron in the hidden layer is denoted by W_{ij} . The sigmoidal response function used is

$$f_B(u) = \frac{1}{1 + \exp(-2\beta u)} \quad (1)$$

Our network receives a 1×24 input vector $X = \{x_k\}; k = 1, 2, \dots, 24$ representing a 16×16 pixel array of a handwritten digit.

The activation u_j of the j^{th} neuron in the hidden layer is

$$u_j = \sum_{k=1}^k w_{jk} x_k \quad (2)$$

The output h_j of the j^{th} neuron in the hidden layer is

$$h_j = f_B(u_j) = \frac{1}{1 + \exp(-2\beta u_j)} \quad (3)$$

Similarly the activation v_i of the i^{th} neuron in the output layer is

$$v_i = \sum_{j=1}^J W_{ij} h_j \quad (4)$$

and the output O_i of the i^{th} neuron in the output layer is

$$O_i = f_B(v_i) = \frac{1}{1 + \exp(-2\beta v_i)} \quad (5)$$

The error between the actual output O and the desired output Y is calculated as follows:

$$E(w) = \frac{1}{2} \sum_{a,j} (y_i^{(\alpha)} - o_i^{(\alpha)})^2 \quad (6)$$

We train the network to minimize the error E over the training set α of 100 examples. The next, we perform the error back propagation algorithm as follows:

- I. Start with random weights
- II. A training vector X_1 is applied to the input. The state of the hidden neurons is determined and then propagated to the output layer where the states O are determined.
- III. We calculate the error from between the actual output O and the desired output Y and modify the weight matrices as follows:

$$\Delta W_{ij} = -\eta \frac{\partial E}{\partial W_{ij}} = \eta (y_i - o_i) f'_i(v_i) h_j \quad (7)$$

$$\Delta w_{ij} = -\eta f'_j(u_j) x_k \sum_q \delta_q W_{qj} = \eta f'_j(u_j) x_k \sum_q (y_q - o_q) f'_q(v_q) w_{qj} \quad (8)$$

- IV. Another training vector X_2 is applied to the network and steps II to III are repeated
- V. Steps II to IV are repeated for all training vectors
- VI. Steps II to V are repeated until the total error E is below a chosen threshold.

The resulting output node with the highest activation is interpreted as the perceived digit. Error back-propagation was used to train the neural network on a training set of 100 digits. The trained network was then tested on a separate set of 20 digits. Some of the parameters explored in the neural network implementation were learning rate, momentum parameter, and training iterations.

6 Parallel Data Fusion Network in Recognition and Classification

Depending on the sensor network topology, it may be more useful to implement the distributed detection or estimation using a tree structure. Tsistsiklis [19] shows that the optimal decision rules are still in the form of threshold tests. Tang *et al.* [20] consider the case where the local decisions made at a number of sensors are communicated to multiple root nodes for data fusion. In the cases discussed above, the

information flows in one direction from the sensors to either the single fusion center or to a number of root nodes. Even in the decentralized market topology, where numerous sensors report to multiple intermediate nodes, the graph of the network is still acyclic. If the communication network is able to handle the increased load, performance can be improved through the use of decision feedback [21][22]. Pados *et al.* [21] examine two distributed structures: 1.) a network where the fusion center provides decision feedback connections to each of the sensor nodes, and 2.) a set of sensors that are fully interconnected via decision feedback. The performance of the fully connected network is quantifiably better, but their initial system was non-robust to variations in the statistical descriptions of the two hypotheses. Robust testing functions are able to overcome this problem, and they show that robust networks tend to reject the feedback when operating with contaminated data. Alhakem and Varshney [22] study a distributed detection system with feedback and memory. That is, each sensor not only uses its present input and the previous fed-back decision from the fusion center, but it also uses its own previous inputs. They derive the optimal fusion rule and local decision rules, and they show that the probability of error in a Bayesian formulation goes to zero asymptotically. Additionally, they address the communication requirements by developing two data transmission protocols that reduce the number of messages sent among the nodes.

Swaszek and Willet propose a more extensive feedback approach that they denote parleying [23]. The basic idea is that each sensor makes an initial binary decision that is then distributed to all the other sensors. The goal is to achieve a consensus on the given hypothesis through multiple iterations. They develop two versions of the algorithm; the first is a greedy approach that achieves fast convergence at the expense of performance. The n th-root approach constrains the consensus to be optimum in that it would match that of a centralized processor having access to all the data. The main issue is the number of parleys (iterations) required to reach this consensus.

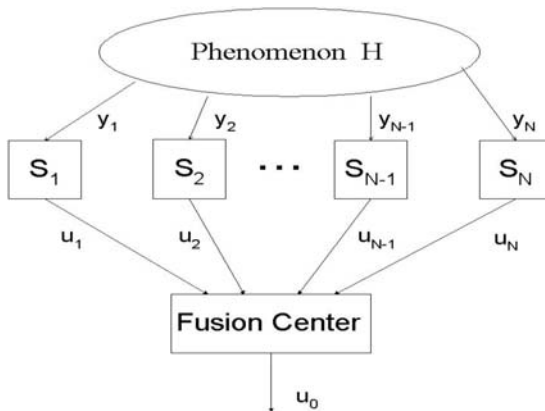


Fig. 4. Parallel data fusion network for multi-sensors

We examine two particular applications of sensor networks in the context of EMS in this paper:

1. Use of BPN to assist the optimal use of temperature, humidity, ultraviolet, and illumination four variable measurements for EMS.
2. Use of evolutionary algorithms to identify and classification the parameters for a control system.

Each intelligent sensor suite consisting of multiple sensors in the overall sensor network is capable of making some decisions based on its own inputs. These decisions are passed on to higher-level nodes in the control hierarchy for information assimilation or information fusion. These decision-making problems can be formulated as hypothesis testing problems in a distributed framework. For optimum results, environmental conditions must be optimized for each and every occupied space based on its particular environmental forcing parameters (e.g., emissions, intensity of light, occupant loads) and the needs of its occupants. The higher fidelity and better resolution information available from a sensor-rich environment will be processed and employed for optimal distributed control. We propose the use of neural networks and evolutionary algorithms to address some of these problems.

Design of Fusion Rules

Input to the fusion center: $u_i, i = 1, 2, \dots, n$

$$u_i = \begin{cases} 0, & \text{if detector } i \text{ decides } H_0 \\ 1, & \text{if detector } i \text{ decides } H_1 \end{cases} \quad (9)$$

Input to the fusion center: u_0

$$u_0 = \begin{cases} 0, & \text{if } H_0 \text{ is decided} \\ 1, & \text{otherwise} \end{cases} \quad (10)$$

Fusion rule: logical function with N binary inputs and one binary output, the number of fusion rules is 2^{2^N}

The possible fusion rules for two binary decisions based on BPN training process. Therefore, the optimum fusion rule that minimizes the probability of error is:

$$\sum_{j=1}^N \left[u_j \log \frac{1 - P_{Mj}}{P_{Fj}} + (1 - u_j) \log \frac{P_{Mj}}{1 - P_{Fj}} \right] \begin{matrix} u_0 = 1 \\ > \log \eta \\ < \log \eta \\ u_0 = 0 \end{matrix} \quad (11)$$

7 Sensor Data Estimation Using Neural Networks

This section discusses the estimation of temperature, humidity, ultraviolet, and illumination four variable measurements in one region of a multi-zone environment using sensor readings obtained elsewhere, with BPN trained for the estimation task. This methodology can be employed for sensors that measure other environmental attributes for data fusion application. Example applications of the problem considered here include:

1. Monitoring large areas using a relatively small number of sensors;
2. Sampling sensors infrequently to save power and communication resources;
3. Being able to operate the control system effectively even when some sensors nodes fail or lose battery power.
4. Detecting possible failures in some sensors nodes using the readings of other sensors and coordinator.

Data Collection

We used a simple experimental test, in which twenty-four wireless nodes, capable of measuring the intensity of temperature, humidity, ultraviolet, and illumination, were placed in each observational point where our campus. Every node point comprises four sensors and a network router. Among four sensors, the fusion center is responsible for analyzing the collected sensing data. These Data from these ninety-six sensors were collected under varying outdoor or indoor environmental conditions, e.g., closing the window blinds, switching off some lights, and at various times of the day. The data collected from the six sensors are shown in Fig. 5. As the numbers of data points are relatively heavy, these data points were duplicated and some noise added to them, to increase the robustness of the training algorithm by BPN.

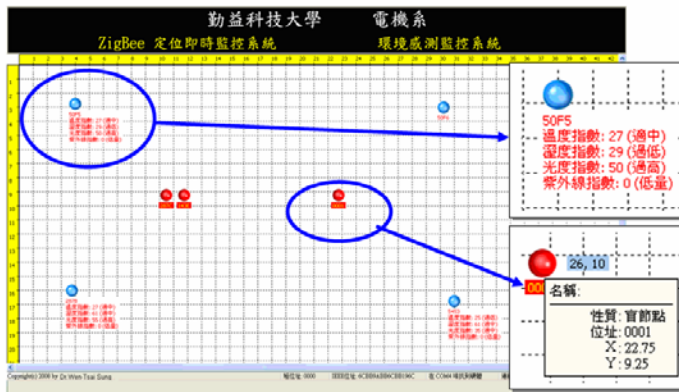


Fig. 5. Experiments result in EMS (estimated place: Yang-Fen automation electrical engineering company at Taichung factory)

The intensity of four type sensors values in the different factory region are different due to different window blind settings, light status and shadowing. Moreover these values change dynamically with different indoor and outdoor environmental changes, e.g., operating a light switch, changing window blind settings, or changing outside light intensity. Since any such change in environmental conditions affects the readings of multiple sensors, we expect that a neural network can be trained to estimate readings at multiple locations based on the observation of a single sensor node.

A one-hidden layer feed forward neural network with sigmoidal node transfer units is the prime candidate for experimentation, since such networks have been frequently used in other function approximation tasks. We also hypothesize that the hidden

nodes in such a network compute features describing environmental conditions, essential to the estimation of light intensities at multiple locations; hence the same hidden nodes can be connected to different output nodes to estimate the readings of multiple sensors. Hence a BPN was applied to estimate the values of three sensors based on the readings of a single sensor.

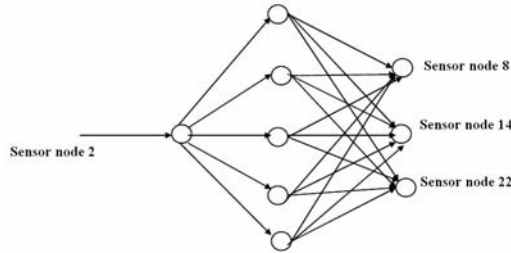


Fig. 6. BPN for estimation of the intensities at neighboring points

The results obtained show that the BPN successfully estimates the intensity of temperature, humidity, ultraviolet, and illumination at three neighboring points based on the intensity at a single point. Estimation accuracy decreases with increasing differences between the environmental conditions at the reference point and at the estimated point.

Data Fusion

Estimations made using a combination of readings from multiple sensors are expected to result in increased success. For this particular application, our simulation results showed that a data fusion model, implemented using a single monolithic BPN module with inputs from multiple sensors, was much less successful than a decision data fusion model in which the estimations made using single sensors are combined using a simple fusion rule, as shown in Figure 7. The absolute error on the training data was 0.04 and the average mean absolute error on the test data was 0.018 after training for 400 hours, demonstrating that decision fusion using BPN modules can give a very good estimate of temperature, humidity, ultraviolet, and illumination intensity values.

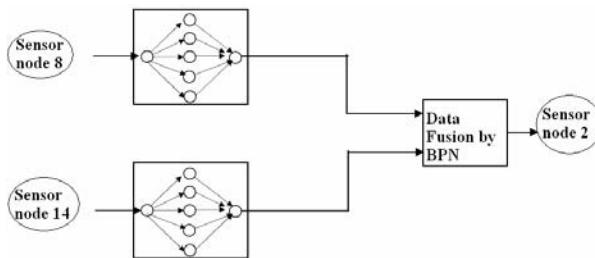


Fig. 7. BPN based decision data fusion model for estimation of the intensity of light at a single point based on the intensities at neighboring points

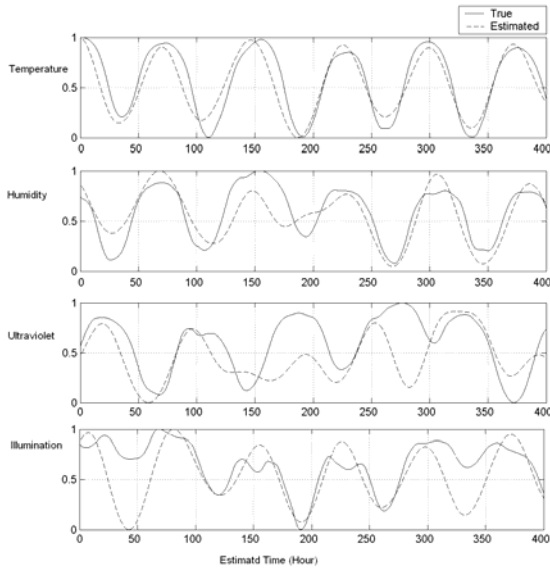


Fig. 8. Results using the data fusion approach: actual vs. estimated values.(Note: Ninety-six Sensors (four type sensors) for Real-time Data Fusion with recognition and classification process via BPN.(Estimated time is 400 hr)).

Performance Improvement Due to Fusion

The WSNs was simulated with four category data from each node and the results for each of the different test conditions for temperature, humidity, ultraviolet, and illumination are given in Fig. 9(a) and Fig. 9(b) respectively. The Fig. 9(a) is non-data fusion process and the Fig. 9(b) indicated it have data fusion process. Traditional data fusion methods compare with innovative BPN methods, the conspicuous distinction is data fusion genuine acceptance rate of the initial stage. This investigates employed BPN with training and learning ability that has improved the data fusion efficient for EMS based on ZigBee WSNs platform. Figure 10 indicated ZigBee WSNs’ Packages analysis and monitoring in this case study refer to our researches.[24]

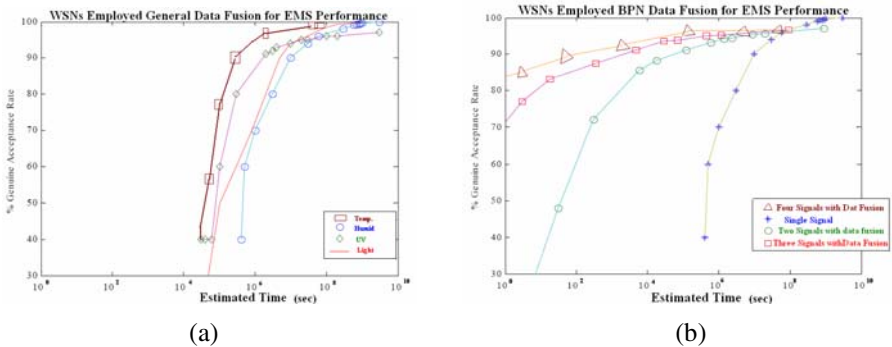


Fig. 9. Results of traditional data fusion methods compare with innovative BPN methods

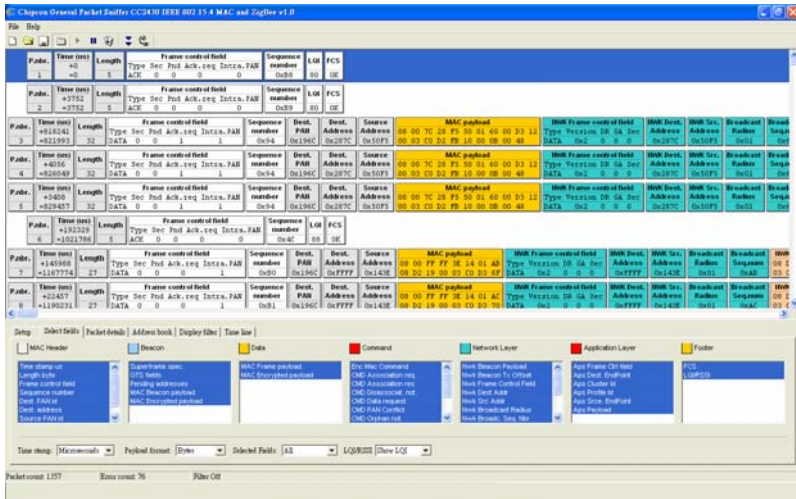


Fig. 10. The ZigBee Network Packages Analysis and Monitoring

8 Conclusions

Sensor networks involve technologies from three related areas: sensing, communication, and computation (hardware, software, and algorithm). Lately a lot of research work has been done in all of these fields to make sensor nodes more intelligent and useful. Wireless Sensor Networks have emerged as a new information-gathering paradigm based on the collaborative effort of a large number of sensing nodes. This study discusses the classification and fusion approach in WSNs, which BPN based feature extraction method is proposed. This method partitions the frequency band in different resolution to distinguish the difference in low-frequency band and reduces the feature dimensions greatly. The extracted feature expresses stable classification rate for different moving condition. Due to the multiresolution property of wavelet decomposition can not only eliminate unstable variety of frequency feature, but also merge discrepancies. Therefore, weighted BPN classification rule uses the distance of feature x and its nearest neighbors to denote the degree committing to each class. This can provide more information of x and its neighbors than that of voting BPN rule and keeps the merits of non-parametric and easy to use.

In this study, we design four various sensors in a circuit board which aggregated temperature, humidity, ultraviolet, and illumination measurements for EMS around at the same time. Our primary network architecture consists of 24 nodes and beacon behavior in real-time mode with interval about 0.5 sec. Multilayer perceptrons BPN have been applied successfully to solve some difficult and diverse problems by training them in a supervised manner with a highly popular algorithm known as the back-propagation algorithm. This algorithm is based on the error correcting learning rule. Finally, the sink nodes would process various signal sources for data fusion in recognition and classification via BPN technology.

Acknowledgement

The authors would like to thank the National Chin-Yi University of Technology, Taiwan for financially supporting this research.

References

- [1] Akyildiz, F., et al.: Wireless Sensor Networks: A Survey. *Journal of Computer Networks* 38(4), 393–422 (2002)
- [2] Franceschini, F.: A review of localization algorithms for distributed wireless sensor networks in manufacturing. *Journal of Computer Integrated Manufacturing* (June 13, 2007)
- [3] Aquino, A.L.L., Figueiredo, C.M.S., Nakamura, E.F., Buriol, L., Loureiro, A.A.F., Fernandes, A.O., Coelho, C.J.N.: Data streambased algorithms for wireless sensor networks. In: *IEEE AINA 2007, Niagara Falls, Canada*, pp. 869–876 (2007)
- [4] Brouwer, R.K.: An integer recurrent artificial neural network for classifying feature vectors. *International Journal of Pattern Recognition and Artificial Intelligence* 14(3), 335–339 (2000)
- [5] Brouwer, R.K.: A fuzzy recurrent artificial neural network for pattern classification. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 8(5), 525–538 (2000)
- [6] Hu, H., Lin, X., Wu, M.-Y.: Multi-Source Data Fusion and Management for Virtual Wind Tunnels and Physical Wind Tunnels. *Autonomous Systems – Self-Organization, Management, and Control*, 63–70 (2008)
- [7] Loskiewicz-Buczak, A., Uhrig, R.E.: Aggregation of evidence by fuzzy set operations for vibration monitoring. In: *Third International Conference on Industrial Fuzzy Control and Intelligent Systems, IFIS apos 1993, December 1993*, vol. 1(3), pp. 204–209 (1993)
- [8] Sharples, S., Callaghan, V., Clarke, G.: A multi-agent architecture for intelligent building sensing and control. *International Sensor Review Journal* (1999)
- [9] Varshney, P.K.: *Distributed Detection and Data Fusion*. Springer, Heidelberg (1997)
- [10] Varshney, P.K., Mohan, C.K.: On Sensor Networking and Signal Processing for Smart and Safe Buildings. In: Szymanski, B.K., Yener, B. (eds.) *Advances in Pervasive Computing and Networking*, pp. 213–226 (2005)
- [11] Qi, H.R., Iyengar, S.S., Chakrabarty, K.: Multiresolution data integration using mobile agents in distributed sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 31(3), 383–391 (2001)
- [12] Kumar, R., Wolenetz, M., Agarwalla, B., Shin, J., Hutto, P., Paul, A., Ramachandran, U.: DFuse: A framework for distributed data fusion. In: *Proceedings of the First International Conference on Embedded Networked Sensor Systems*, pp. 114–125. ACM Press, Los Angeles (2003)
- [13] Zhao, F., Liu, J., Liu, J.J., Guibas, L., Reich, J.: Collaborative signal and information processing: An information directed approach. *Proceedings of the IEEE* 91(8), 1199–1209 (2003)
- [14] Jayasimha, D.N., Iyengar, S.S., Kashyap, R.L.: Information integration and synchronization in distributed sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics* 21(5), 1032–1043 (1991)
- [15] Polikar, R., Udpa, L., Udpa, S.S., Taylor, T.: Frequency Invariant Classification of Ultrasonic Weld Inspection Signals. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control* 45(3), 614–625 (1998)

- [16] Polikar, R., Udpa, L., Udpa, S.S., Spanner, J.: Time Scaling and Frequency Invariant Multiresolution Analysis of Ultrasonic NDE Signals. In: Thompson, D.O., Chimenti, D.E. (eds.) *Review of Progress in Quantitative NDE*, vol. 17, pp. 743–749. Plenum Press, New York (1998)
- [17] Bae, S., Udpa, L., Udpa, S.S., Taylor, T.: Classification of Ultrasonic Weld Inspection Data Using Principal Component Analysis. In: Thompson, D.O., Chimenti, D.E. (eds.) *Review of Progress in Quantitative NDE*, vol. 16, pp. 741–748. Plenum Press, New York (1997)
- [18] Ye, Y.C.: *Application and Implementation on Neural Network Models*. Scholars Books Co., Ltd. (2004)
- [19] Tsistsiklis, J.N.: Decentralized detection. In: Poor, H.V., Thomas, J.B. (eds.) *Advances in Statistical Signal Processing, Signal Detection*, vol. 2. JAI, Greenwich (1993)
- [20] Tang, Z.B., Pattipati, K.R., Kleinman, D.L.: Optimization of distributed detection networks: Part II generalized tree structures. *IEEE Trans. Syst., Man Cybern.* 23, 211–221 (1993)
- [21] Pados, D.A., Halford, K.W., Kazakos, D., Papantoni-Kazakos, P.: Distributed binary hypothesis testing with feedback. *IEEE Trans. Syst., Man and Cybern.* 25, 21–42 (1995)
- [22] Alhakeem, S., Varshney, P.K.: Decentralized Bayesian hypothesis testing with feedback. *IEEE Trans. Syst., Man Cybern.* 26, 503–513 (1996)
- [23] Swaszek, P.F., Willett, P.: Parley as an approach to distributed detection. *IEEE Trans. Aerospace Elect. Syst.* 31, 447–457 (1995)
- [24] Sung, W.-T., Chung, H.-Y.: Design an Innovative Localization Engines into WSN via ZigBee and SOC. In: 2008 CACS International Automatic Control Conference (2008)

Argus: A Light-Weighted Secure Localization Scheme for Sensor Networks

Wen Tao Zhu¹ and Yang Xiang²

¹ State Key Laboratory of Information Security,
Graduate University of Chinese Academy of Sciences, Beijing 100049, China
wtzhu@ieee.org

² School of Management and Information Systems, Central Queensland University,
North Rockhampton, Queensland 4702, Australia
y.xiang@cqu.edu.au

Abstract. Rapid technological advances have enabled the development of wireless sensor networks for various monitoring tasks. Usually the involved applications are dependant on the location knowledge of the low-cost sensor nodes, the majority of which are non-beacon nodes whose positions are yet to be discovered. We present Argus, a light-weighted position estimation scheme for sensor networks, to address the problem of localizing non-beacon nodes, with the location references obtained from a few beacon nodes whose positions are known apriori. We first determine certain geometric reference points and evaluate them with a voting procedure. Then the position of a non-beacon node of concern is estimated with the geometric centroid of the identified most valuable reference points. Simulation results show that even when a few of the available beacon nodes are malfunctioning, our scheme can tolerate those misleading location references, and still provide a dependable localization service.

Keywords: Wireless sensor network, beacon and non-beacon nodes, secure localization, fault-tolerance, dependable computing.

1 Introduction

Recent advances in electronics and wireless communications have made it possible to deploy tiny-size, low-cost, and multifunctional wireless sensor nodes in a large-scale field to monitor the ambient conditions such as temperature, humidity, and pressure, in commercial, residential, and military areas. These sensor nodes are designed to monitor and report local states and events in their vicinities, and a large collection of such nodes form a wireless sensor network (WSN) in a distributed and self-organizing manner [1]. The WSN has emerged as an important and also economic means for monitoring the physical world, and it can be used for various applications like emergency response, energy management, environmental and medical monitoring, logistics and inventory management, and battlefield surveillance [2].

A sensor node is usually made up of basic components like sensing, data processing, transceiver, and power units. It is also common that a sensor node has a location finding system [1], i.e., so-called localization system [3] [4]. The positions of sensor nodes do not necessarily need to be engineered or predetermined, which facilitates random deployment in inaccessible terrains or disaster relief operations (e.g., by dropping nodes from a flying helicopter). Nevertheless, in many applications (e.g., search and rescue, wildlife monitoring, and target tracking), it is assumed that after the deployment of the WSN, each sensor node can discover or estimate its real position. In other words, sensor nodes are generally supposed to be *location aware* so as to fulfill relevant monitoring tasks. In this paper, the notions position estimation and location discovery are used interchangeably.

The importance of sensor localization arises from several factors [3] [4]. Consider applications like building/forest monitoring, where ambient conditions must be perceived and passed on in a certain integrated manner. It tends to be meaningless if a sensor merely signals an event like “smoking”, as in such case it is impossible for actions to be immediately taken. Only when the detection is combined with the position of the reporting sensor (i.e., the origin of the event) can we be clearly aware of what is exactly taking place, e.g., where a fire is about to start. In general, knowledge of sensor locations can significantly assist in many functions like ad hoc routing, network management (e.g., congestion control), data-centric storage, and key establishment. Therefore, localization plays an essential role for the development and operation of WSNs.

Recently, quite a few localization schemes have been proposed in the literature [5, 6, 7, 8, 9, 10], which typically estimate sensor positions by somehow solving mathematical optimization problems with certain constraints. However, they generally assume that the location references obtained from benchmark sensors known as beacon nodes are all reliable, which may not always be the case. For example, a beacon node mounted on the ceiling of a hall may fall off during an earthquake, or one in a forest sensor network may be moved off its place due to heavy rain or a curious animal. For another example, sensor nodes are typically deployed in unattended, even hostile regions, and are not made tamper-proof due to cost considerations; besides the fact that sensors are subject to random failures, it is not unusual that a malicious adversary can compromise some of them. Malfunctioning beacon nodes may cause significant positioning confusion, and directly degrade the performance of basic functions like tracking, routing, and key establishment, and thus may affect many monitoring tasks.

Note that in the presence of capture and compromise, cryptographic techniques like authentication and encryption cannot effectively protect WSN localization against attacks; indeed, many cryptographic mechanisms [11, 12, 13] themselves are based on the assumption that sensor deployment knowledge is available and trustworthy. Therefore, most localization algorithms use non-cryptographic security techniques, and only rely on cryptography as a second line of defense [4].

In this work, we propose a novel localization scheme for low-cost sensor nodes focusing on service dependability. The design goal is to tolerate a few malfunctioning benchmark (i.e., beacon) nodes, be the malfunction caused by random

Table 1. Notation

B_i	a beacon node, whose static position is claimed to be at (x_i, y_i)
N	the non-beacon node of concern, whose position (x, y) is unknown
(x_i, y_i, d_i)	the location reference from a beacon node B_i
O_i	the reference circle introduced by B_i
$P(i, j, k)$	a reference point generated by O_i and O_j , where k is either 1 or 2
$v(i, j, k)$	number of votes received by $P(i, j, k)$
a	the deviation introduced by a malfunctioning beacon node
c	number of malfunctioning beacon nodes
e_i	the unavoidable physical measurement error regarding (x_i, y_i, d_i)
e_{max}	the maximum measurement error in a given sensor deployment field
e_N	the location estimation error of a certain algorithm
l	the side length of the sensor deployment field
n	number of available beacon nodes
p_s	the probability for our scheme to switch to the survival mode
r	the transmission range of a wireless sensor node

failures or malicious attacks. For readers' convenience, the symbols and parameters employed through the paper are summarized in Table 1.

The rest of this paper is organized as follows. Section 2 introduces involved terminology and necessary assumptions, as well as some related work. Section 3 presents Argus our light-weighted secure localization scheme, which is based on some rudimentary geometric principles and a voting process. In Section 4 we present and discuss our simulation results. Section 5 concludes the paper.

2 Localization in Sensor Networks

2.1 Beacon and Non-beacon Nodes

A sensor network consists of wirelessly connected nodes usually diversely released to perform various monitoring tasks, where location awareness is inherently one of the most essential system functionalities. In the typical approach, a small percentage (due to cost concerns) of the sensors known as *beacon nodes* are aware of their positions, e.g., with manual placement or external means like GPS [3], which serve as the basis for WSN localization. Accordingly, sensors whose positions are yet to be discovered are called *non-beacon nodes*. The location finding system [1] is designed to allow the non-beacon nodes to estimate their physical positions [3], employing the known locations of the beacon nodes.

Location discovery protocols [5, 6, 7, 8, 9, 10] usually work as follows. After the WSN deployment, the beacon nodes broadcast radio signals to their neighbors, where their own positions are capsulated in the beacon packets. Non-beacon nodes can then measure the received beacon signals to estimate their own positions. Again note that although some protocols like [5] adapt to the addition or death of sensors, they generally assume all beacon nodes are *benign* ones, while possibilities of beacon packets containing misleading payload from *malfunctioning* beacon nodes (due to either random accidents or deliberate attacks) are overlooked.

2.2 Range-Based Localization

Localization protocols can be divided into range-based and range-free [7, 8]. The former employs absolute point-to-point position estimates, while the latter only needs the existences of beacon signals for coarse-grained location discovery (e.g., distances between nodes are only counted in hops). In this work we address the former. For simplicity we consider a static sensor network. Usually all the sensor nodes are assumed to be randomly distributed on a two-dimension plane, and such a model has been generally adopted in the literature [3].

We assume the information that a non-beacon node N obtains from a beacon node B_i , referred to as a *location reference*, contains: (i) (x_i, y_i) the known position declared by B_i , and (ii) d_i the distance between the two nodes locally measured by N from certain physical features of the wireless beacon signal [3], like received signal strength indicator (RSSI), time (difference) of arrival (ToA/TDoA), and angle of arrival (AoA) [6]. Note that even in AoA methods, N can still derive the distances to the beacon nodes [6].

We assume all sensor nodes have the same transmission range r and communicate with each other via bi-directional wireless links, and define a location reference as the triple (x_i, y_i, d_i) , where $d_i (< r)$ is the distance measured by N to B_i claiming at (x_i, y_i) . When N has obtained enough location references, it can then estimate its own position. Note that although d_i 's are physically gauged from beacon signals and only contain predictable measurement errors, the positions of B_i 's are broadcast to N as payload information in the beacon packets. Therefore, even if cryptographic technologies are employed, it is still possible for malfunctioning beacon nodes to declare arbitrary locations, intentionally or unintentionally.

2.3 Minimum Error Estimation

A typical localization approach is to regard the location references as constraints that a non-beacon node's position must satisfy, and the position is estimated by finding a mathematical solution that meets the constraints with minimum estimation error. As depicted in Fig. 1, in a Euclidean plane the position of an arbitrary point (the non-beacon node N) can be uniquely determined by the distances from at least three non-collinear known points (beacon nodes B_1 , B_2 , and B_3). Now we outline the Minimum Mean-Square Error Estimate (MMSEE) method, which have been adopted in most range-based and even some range-free localization protocols [5, 6, 7, 8, 9, 10].

We employ Fig. 1 to illustrate the *trilateration* localization method. Three beacon nodes B_1 , B_2 , and B_3 are preloaded with known positions, which are then broadcast via beacon signals. We refer to the virtual circle O_i centered at $B_i(x_i, y_i)$ with radius d_i as a *reference circle*. Now a non-beacon node N needs to find its location (x, y) , which theoretically is the intersection of reference circles O_1 , O_2 , and O_3 . To discover its position, N measures the distances d_i 's to the beacon nodes B_i 's. In real-world applications, due to the inaccuracy introduced by the physical measurement of wireless signals, the associated *measurement*

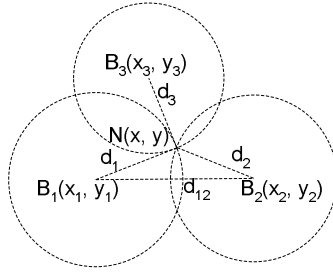


Fig. 1. The position (x, y) of a non-beacon node N can be determined according to its distances from at least three beacon nodes, each of which declares its known position (x_i, y_i) in the broadcast beacon signal. Theoretically, N is positioned at the intersection of all the reference circles.

error is unavoidable [3]. For a location reference (x_i, y_i, d_i) and N actually positioned at (x_0, y_0) , we define

$$e_i \stackrel{def}{=} \sqrt{(x_0 - x_i)^2 + (y_0 - y_i)^2} - d_i \tag{1}$$

as the measurement error. In many cases, instead of e_i or $-e_i$ we only care the absolute value $|e_i|$, and we assume in a given WSN it is bounded by $|e_i| \leq e_{max}$ for any i .

In Fig. 1, with location references $(x_i, y_i, d_i)_{i=1,2,3}$, non-beacon node N can discover its position (x, y) by finding solutions x and y that minimize the total squared error E between calculated Euclidean distances and measured distances:

$$E \stackrel{def}{=} \sum_{i=1}^n (\sqrt{(x - x_i)^2 + (y - y_i)^2} - d_i)^2. \tag{2}$$

In Fig. 1, $n = 3$ as there only exist three location references. Usually quite a few (e.g., a dozen of) location references are available, and they help improve the estimation accuracy. This corresponds to the more general case of trilateration known as *multilateration*. The minimum value of the total squared error E is then achieved when both partial derivatives of E reach 0, and thus (x, y) can be computed by solving a set of equations:

$$\begin{cases} 0 = \frac{\partial E}{\partial x} = 2 \sum_{i=1}^n (x - x_i) \left(1 - \frac{d_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}}\right) \\ 0 = \frac{\partial E}{\partial y} = 2 \sum_{i=1}^n (y - y_i) \left(1 - \frac{d_i}{\sqrt{(x - x_i)^2 + (y - y_i)^2}}\right) \end{cases}. \tag{3}$$

One concern of this method is that solving the equations may incur much computational overhead, which is undesirable for low-cost sensors. Moreover, it is not so intuitive to apply the MMSEE method to *secure localization*, as it does not provide proper context for dependable computing when malfunctioning beacon nodes may declare misleading positions.

3 Localization Based on Reference Points

3.1 Preliminaries

The security of sensor localization can be partially protected by cryptographic technologies. It may be necessary to employ broadcast authentication schemes like μ TESLA [2] or BABRA [13], such that the non-beacon nodes can assure that the beacon packets are sent from the authorized beacon nodes. Nevertheless, authentication alone cannot guarantee localization security. For example, as sensor nodes are not made tamper-proof, an attacker may compromise a beacon node to acquire the cryptographic materials so as to craft authenticated but incorrect information. For another example, the attacker may replay legitimate beacon packets, including employing wormhole attacks [13] where packets are tunnelled from one place to another distant place, so as to skew a non-beacon node's view of its surrounding beacon nodes. These crafted or replayed packets bear proper authentication information and they can pass cryptographic checks. Therefore, to secure location discovery in sensor networks, additional mechanisms are pursued, which particularly are not explicitly dependant on cryptographic key management. Moreover, for low-cost sensor nodes, light-weight algorithms are preferred.

We assume certain broadcast authentication mechanism [2] [13] is readily available, and we consider the case that a malfunctioning beacon node B_i may mislead a non-beacon node N into accepting incorrect location reference (x_i, y_i, d_i) . To do so, B_i can either declare (intentionally or unintentionally) a wrong position (x_i, y_i) in the beacon packet, or manipulate the physical features of the beacon signal (e.g., alter the radio transmission power in RSSI based technique) such that the measurement by N results in an abnormal d_i . Note that in the latter case, the signal should be neither too strong (otherwise such manipulation will be easily detected by nearby sensor nodes, either beacon or non-beacon) nor too weak (otherwise the signal may not reach an intended non-beacon node). As these two attack strategies are similar in the sense of introducing a misleading location reference (x_i, y_i, d_i) , in this paper we only consider the former.

Regarding its own position, a malfunctioning beacon node B_i may introduce arbitrary deviations into the broadcast coordinates x_i and y_i , and thus mislead a non-beacon node N into making a very poor estimation. However, we assume that one distinct beacon node only declares one location; otherwise it is trivial for N to detect the malfunction based on inconsistent behaviors of a broadcast-authenticated B_i . Therefore, whether the incorrect declaration (x_i, y_i) is due to random failures or malicious attacks, it has to be unique for B_i .

3.2 Reference Points

Based on some rudimentary geometry, we propose Argus our localization protocol as a light-weighted alternative to the traditional MMSEE approach. Our design goal is to withstand a few malfunctioning beacon nodes and to provide a dependable position estimation service. The underlying idea is that, in the ideal case, a reference circle has two intersections with each of other circles (refer

Fig. 1). These intersections, known as *reference points* in our algorithm, serve as good indications for the position of the non-beacon node of concern. When there are a dozen of beacon nodes, there can be around 100 reference points, and thus we name the scheme after Argus, the hundred-eyed giant in Greek mythology.

Assume there are n available beacon nodes, most of which are benign ones supplying correct location references while the rest a few may be malfunctioning. Recall that in Fig. 1, it is implied that when all beacon nodes are benign and there are no measurement errors, N the non-beacon node of concern is theoretically positioned at the common intersection of all the n reference circles ($n=3$ in Fig. 1). Due to measurement errors, in reality there does not exist such a common intersection. However, we can still expect that N is very close to a certain set of reference points. The key idea here is to treat the points selectively, and to harness the observation that N should be close to the most “valuable” reference points.

For example in Fig. 2(a) where there are merely two beacon nodes B_1 and B_2 , we can only expect the non-beacon node to be located near one of the two reference points N_1 and N_2 , which are the intersections of reference circles O_1 and O_2 . In this case we say points N_1 and N_2 are equally valued; neither is “valueless”. Assume then a third beacon node B_3 provides the additional information of reference circle O_3 , due to the existence of which N_1 and N_2 are no longer equally valuable. With certain criterion we can decide whether N_1 or N_2 is the more valuable one, as the case illustrated in Fig. 2(b) or Fig. 2(c), respectively. At the same time, the third circle O_3 introduces four more reference points, among which similarly some are more valuable than the others. Now that there are three beacon nodes, there can be up to six reference points, and only a portion of them are regarded the most valuable.

For n reference circles, there may be $2 \cdot \binom{n}{2} = n(n - 1)$ reference points at the most. We gauge the value of each of them with a voting process concerning simple geometric constraints, and the value of a reference point is quantified with the number of votes it receives. Then we estimate the position of the non-beacon node based on the most valuable reference points, which have got the most number of votes. For example, in Fig. 2(b) there are only two such most valuable reference points, while in Fig. 2(c) there are three.

For simplicity we omit the rare possibility that two reference circles are tangent. Hence a reference circle O_i has either two or zero intersections with

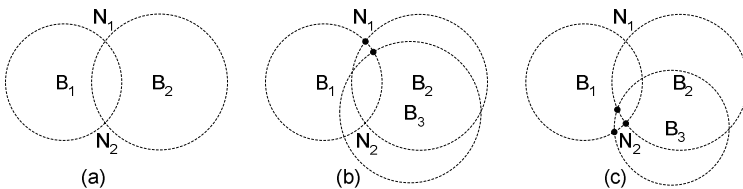


Fig. 2. Intersections of reference circles serve as good indications: The non-beacon node’s position is expected to be close to those most “valuable” reference points, where there are two in case (b), and three in case (c)

any other O_j , $1 \leq i < j \leq n$. If they have two intersections, we denote the one with a less x value (in the horizontal axis) as $P(i, j, 1)$, and the other as $P(i, j, 2)$. In case both intersections are in a vertical line, we denote the one with a less y value as $P(i, j, 1)$, and the other as $P(i, j, 2)$. Note that $P(i, j, k) = P(j, i, k)$ for any $i \neq j$ and $k = 1, 2$. To gauge the value of any reference point $P(i, j, k)$, $1 \leq i < j \leq n, k = 1, 2$, we associate it with an integer variable $v(i, j, k)$ known as the voting counter. The counter is initialized to 0 and is incremented by 1 each time the associated reference point receives a positive vote.

3.3 Voting the Reference Points

Our approach for voting the reference points can be simply implemented in a sensor node which only supports basic arithmetic operations. To begin with, we employ a function d^2 to compute the square of the Euclidean distance between any two points $A_1(x_1, y_1)$ and $A_2(x_2, y_2)$:

$$d^2(A_1, A_2) \stackrel{def}{=} (x_1 - x_2)(x_1 - x_2) + (y_1 - y_2)(y_1 - y_2), \quad (4)$$

which can be easily implemented in any kind of low-cost processor that supports float-point addition and multiplication. We prefer it to be as light-weighted as possible because d^2 will be frequently invoked in Argus as a basic primitive.

We have each reference circle “vote” on the reference points’ values. As one may expect, herein a voting counter $v(i, j, k)$ does not count votes from reference circles O_i and O_j themselves, as actually a beacon node $B_l(x_l, y_l)$ unconditionally trusts all the points positioned on its reference circle O_l . Given a reference point $P(i, j, k)$ and a beacon node B_l with location reference (x_l, y_l, d_l) , $1 \leq l \leq n, l \neq i, l \neq j$, the non-beacon node N calculates the distance d between $P(i, j, k)$ and B_l . If d is between $\max(0, d_l - e_{max})$ and $(d_l + e_{max})$, we say $P(i, j, k)$ complies with the location reference from B_l , and increase the voting counter $v(i, j, k)$ by 1. Actually, the real-world condition is carried out as follows (though e_{max} the maximum measurement error is not rendered in Fig. 2):

$$(\max(0, d_l - e_{max}))^2 < d^2(P(i, j, k), B_l) < (d_l + e_{max})^2. \quad (5)$$

As each $P(i, j, k)$ is associated with a $v(i, j, k)$, $1 \leq i < j \leq n$ and $k = 1, 2$, the non-beacon node N estimating its position needs to maintain $n(n - 1)$ voting counters at the most, each of which can be implemented with just one byte. As each $v(i, j, k)$ receives $(n - 2)$ votes from each B_l ($1 \leq l \leq n, l \neq i, l \neq j$), 1 for positive and 0 for negative, the condition (5) is totally checked $n(n - 1)(n - 2)$ times at the most, i.e., the primitive d^2 depicted in (4) is invoked $n(n - 1)(n - 2)$ times at the most. As N can only be reached by the beacon nodes residing within N ’s communication range r , usually n is relatively small. Therefore, both the storage and the processing costs for N are acceptable. Particularly, simulations show that Argus even works with as few as only 4 benign beacon nodes.

Algorithm 1. Voting the Reference Points

```

for (i=1; i<=n-1; i++) {
  for (j=i+1; j<=n; j++) {
    if (intersecting((0i, 0j) == false) continue;
    ... //two intersections P(i,j,1) and P(i,j,2)
    for (k=1;k<=2;k++) {
      v(i,j,k) = 0;
      for (l=1; l<=n; l++) {
        if (l==i||l==j) continue;
        D=d2(P(i,j,k),Bl);
        if (max(0,dl-emax)*max(0,dl-emax)<D
            && D<(dl+emax)*(dl+emax)) v(i,j,k)++;
      }//l
    }//k
  }//j
}//i

```

We illustrate the voting algorithm in C-like pseudo code in Alg. 1, where the keyword `continue` is employed to skip the current execution within a loop. Note that in Alg. 1, when computing the intersections it is feasible to replace the involved square root operation with a cost-efficient alternative based on Newton's method; we do not dwell on the details due to page limit. When all these votes ($n(n-1)(n-2)$ at the most) are done, we single out the most valuable reference points which have received the most votes. We then choose the geometric centroid of these points as N 's estimated position.

3.4 Computing the Geometric Centroid

The algorithm depicted in Fig. 2 and Alg. 1 outputs an array of reference points that are found to be the most valuable. The geometric centroid of such a finite set of points can be easily computed with the arithmetic mean regarding each coordinate of the points. For example, if only two points are left as the case in Fig. 2(b), the centroid will be their middle point; if three points are left as with Fig. 2(c), the centroid will be the intersection of the three medians of the triangle determined by the three most valuable points, i.e., the triangle's barycenter.

An earlier approach computing a similar geometric centroid is [14], which interestingly employs bounding boxes (instead of reference points). The scheme first determines the intersection of all n bounding boxes, which turns out to be a restricted rectangle area, and then chooses the center of the rectangle as the discovered location for the non-beacon node. However, the bounding box method incurs greater estimation error than trilateration/multilateration approaches [3], on which our algorithm is based. Another approach involving the geometric centroid is found in [7], but due to its range-free nature, it is intended only for applications with less required precision. One of the two schemes proposed in [10] also employs a geometric centroid-based position estimation, where the entire WSN deployment field is

quantized into a grid of square cells, and each beacon node votes on those cells in which the non-beacon node of concern may reside. Then the scheme chooses the centroid of the most valuable cells as the estimated position. Herein the cell [10] is just like the bounding box in [14], and to refine the estimation to a preferred granularity, the entire WSN deployment field has to be partitioned into very small (and thus a very large number of) cells.

At this final step, however, instead of outputting the estimated position of the non-beacon node, Argus may occasionally fail with this *regular* mode depicted in Fig. 2 and Alg. 1, which yields no reference points at all (and thus no centroid). This may be caused by a malfunctioning beacon node, which introduces so much deviation in its claimed position that the associated reference circle is simply too far away from other circles and thus it has no intersection with others. This may also be attributed to any two benign beacon nodes, due to the measurement errors in whose location references the two reference circles have no intersections (though they are very close). Therefore, as a whole it is possible for the n reference circles to turn out to have no intersection. When there are no such regular reference points, Argus switches to the *survival* mode: we connect the two centers of each pair of reference circles, and specify the two (of four) adjacent intersections of the line with the two circles as reference points.

We denote the probability for Argus to turn to the survival mode by p_s . According to the above analysis, p_s only becomes detectable when n is quite small, as in this case there are fewer reference circles available and thus they might yield no intersections. From extensive simulations, we find $p_s \approx 0.2\%$ in the worst case, where n is as few as 4. Note that $n = 4$ makes many existent localization algorithms not function well.

4 Simulation Results

We now present the simulation results of Argus our light-weighted secure localization scheme based on reference points. We focus on performance evaluations regarding estimation accuracy under various conditions. In all simulations, the target field of interest is instantiated in the view of a particular non-beacon node N , and is modelled as a circle with radius r centered at N . We assume there are n randomly deployed beacon nodes B_1, B_2, \dots, B_n located within this circle, and thus their beacon signals can all reach N . For convenience we set the origin of the coordinate axes at the position of node N . We model the entire WSN deployment field as an $l \times l$ square area centered at N , where the side length l is much larger than r . As the target field of interest should be a very limited region of the entire deployment field, we assume n is relatively small so as to comply with the view of node N . In all simulations, we set $r = 50\text{m}$, $l = 600\text{m}$, and $n = 11$.

Of all the n beacon nodes, we assume the first c nodes are malfunctioning, while the rest $(n - c)$ nodes are benign. For example, Fig. 3 depicts a target field with $n = 11$, where there are $c = 4$ malfunctioning beacon nodes (marked as solid dots) and $n - c = 7$ benign ones (represented with hollow dots). Besides N , there

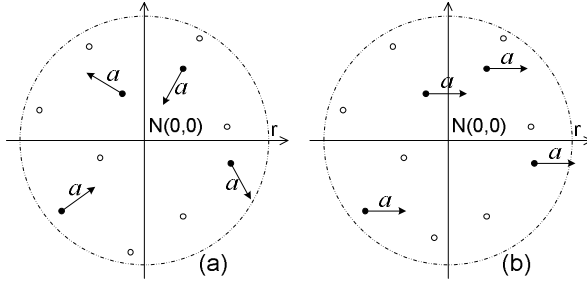


Fig. 3. In the view of the non-beacon node N , the target field is within a circle of radius r centered at itself, containing n randomly deployed beacon nodes. In the first malfunctioning scenario, each of the c randomly failed beacon nodes declares a position a meters away from its real location in an arbitrary direction. In the second scenario, each compromised beacon node increases its horizontal coordinate by $\Delta x = a$ meters.

may also be many other non-beacon nodes in the field, which we do not care. For N 's measurement error as defined in (1), we assume a simple distribution model as $e_i \in_R [-e_{max}, e_{max}]$. That is, each e_i is independently, randomly, and uniformly distributed between $-e_{max}$ and e_{max} .

We refer to the random deployment of the beacon nodes B_1, B_2, \dots, B_n as a *beacon layout*. Although the target field is instantiated concerning the non-beacon node N , we note that the beacon layout is irrelevant to any view of the non-beacon nodes, particularly, the view of N . Therefore, in our simulations the deployment position (x_i, y_i) of a beacon node B_i is generated as follows (i begins with 1): We select a candidate point (x_i, y_i) , where $x_i, y_i \in_R [-\frac{l}{2}, \frac{l}{2}]$, and check whether the distance between $B_i(x_i, y_i)$ and $N(0, 0)$ is within r . If not, we discard such a candidate point and select a new one for the distance test; otherwise, we record such a B_i and yield B_{i+1} in the same manner, until all n beacon nodes are generated. Note that Fig. 3(a) and Fig. 3(b) employ the same beacon layout just to facilitate the illustration of different malfunction scenarios (explained later), while this is not necessarily the case in our simulations. In fact, we simulate Argus on a Linux platform with each beacon layout randomly generated, and it is believed that in all our simulations, none of the beacon layouts ($\sim 10^6$ in all) is re-employed. For each combination of the parameters (like c the number of malfunctioning beacon nodes and e_{max} the maximum measurement error), we run the simulation 10^4 rounds (each employs a unique beacon layout) to draw statistics.

For the estimated position (x, y) of N which actually resides at $(0, 0)$, we investigate $e_N = \sqrt{x^2 + y^2}$ the *estimation error*, in an average sense from the 10^4 rounds of the simulation with a certain parameter combination. Intuitively, e_N indicates how far away the estimated position is from N 's real deployment location. We expect our secure localization algorithm to withstand a few malfunctioning beacon nodes, and to provide the localization service in a *dependable* manner, i.e., e_N is only on the order of the maximum measurement error e_{max} .

We consider two different malfunctioning scenarios, which resemble the two cases that the malfunction is caused by random failures and that caused by malicious attacks, respectively. In the first case, the c malfunctioning beacon nodes are non-colluding; each B_i actually at (x_i, y_i) claims a wrong position $(x_i + \Delta x, y_i + \Delta y)$ on its own that is a meters away from its real location, in an unorganized and random manner as depicted in Fig. 3(a). This is modelled as $\Delta x = a \cos \varphi$ and $\Delta y = a \sin \varphi$, where $a > 0$ and $\varphi \in_R [0, 2\pi]$. In the second case, the c malfunctioning beacon nodes collude to declare deceiving locations that appear consistent in themselves. This is implemented by setting $\Delta x = a$ meters away from each compromised beacon node's actual location ($\Delta y = 0$). In other words, each increases its horizontal coordinate by a meters, in an organized and unified manner ($\varphi = 0$). As depicted in Fig. 3(b), this way they conspire to introduce a deviation in the horizontal axis that is a meters away from N 's real position. In either of the two scenarios, if we model a on the order of e_{max} , the malfunctioning beacon nodes will just serve more or less like the benign ones. In order to have the malfunctions effectively mislead the non-beacon node N , we assume a is on the order of r (but not necessarily on the order of l), particularly, ranging between r and $4r$.

We first present the simulation results regarding the second scenario (colluding attack, $\varphi = 0$). Intuitively, in this case we can only expect Argus to tolerate a

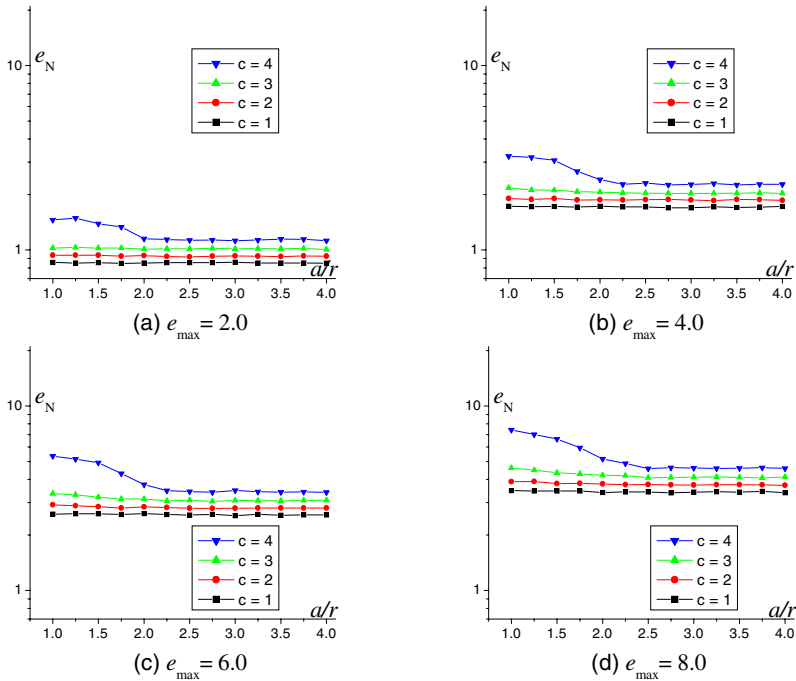


Fig. 4. Average location estimation error e_N in the colluding attack scenario, representing that the malfunction is caused by well coordinated attacks

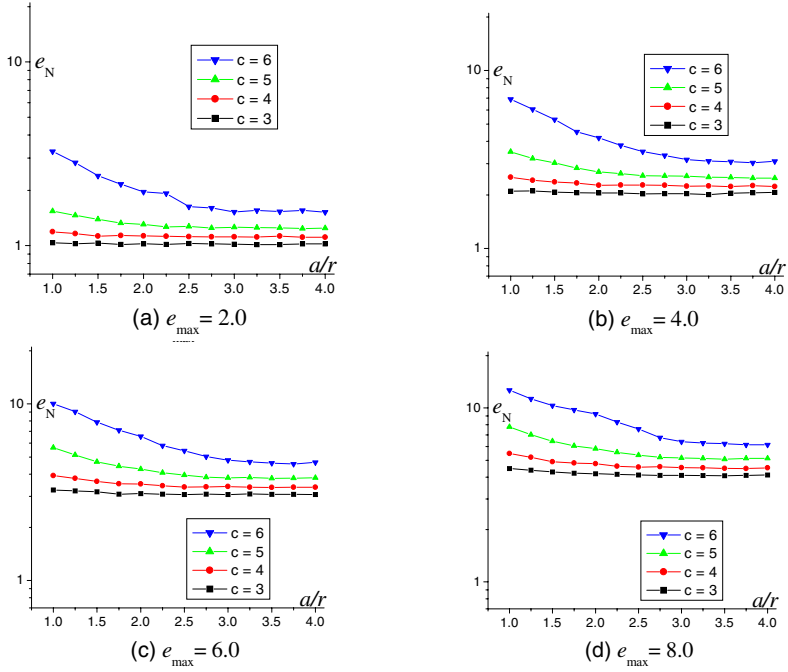


Fig. 5. Average location estimation error e_N in the non-colluding scenario, representing that the malfunction is caused by random sensor node failures

relatively small c the number of compromised beacon nodes, which can be up to 4 according to our results. We plot the average position estimation error e_N under different conditions in Fig. 4. It shows that e_N is obviously affected by (more or less proportional to) e_{max} , but is favorably less than e_{max} . However, when c reaches 5, Argus radically becomes undependable, implying that it cannot tolerate the situation where the attackers have compromised a large portion (nearly half) of the beacon nodes. Note that in Fig. 4 when $a > 2.0r$, e_N tends to be stable; it is subject to e_{max} and c but not to the deviation a . This shows that when $a > 2.0r$, the malicious location references can no more affect the secure localization scheme. Note that this is not the case with MMSEE-based localization, as even if there is only $c = 1$ compromised beacon node, it may introduce a location estimation error, which roughly grows linearly with a , and thus unfortunately, can be arbitrarily large [10].

As to the non-colluding scenario (random failure, $\varphi \in_R [0, 2\pi]$), since this kind of malfunction is unorganized and less vicious, one may expect that under the same parameter combination, Argus should perform better than in the colluding case. Therefore, we are interested with a relatively larger c . Shown in Fig. 5 are the simulation results regarding $c = 3, 4, 5, 6$. Similar to Fig. 4 the colluding attack case, e_N roughly scales with e_{max} , but e_N is always less than e_{max} even at $c = 5$. When $c = 6$, e_N is still on the order of e_{max} , and drops

below e_{max} when the deviation a is large enough ($a \geq 2.5r$). This suggests that Argus is particularly fault-tolerant with random failures, and can still provide a dependable service even if more than half of the beacon nodes encounter such accidents.

5 Concluding Remarks

In many WSN applications it is very important to localize the sensors with a certain position estimation scheme, with the only assistance from a limited number of beacon nodes, whose positions are already known but some may be misleading. Therefore, this work is motivated by the preference that the scheme be not only cost-efficient but also fault-tolerant. We show that by employing certain basic geometric principles, it is feasible to design a light-weighted but secure localization scheme as an improvement over the traditional MMSEE-based localization, which does not provide native support for dependable computing under a challenging environment [10].

The presented scheme is found able to tolerate a small number of malfunctioning beacon nodes and provide dependable position estimation services, be the malfunction caused by organized attacks or random failures. Our future work may include incorporating ideas like intrusion/attack detection in secure localization research as well as field experiments.

References

1. Akyildiz, I.F., Su, W., Sankarasubramanian, Y., Cayirci, E.: A survey on sensor networks. *IEEE Communications Magazine* 40, 102–114 (2002)
2. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: SPINS: Security protocols for sensor networks. *Wireless Networks* 8, 521–534 (2002)
3. Boukerche, A., Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F.: Localization systems for wireless sensor networks. *IEEE Wireless Communications* 14, 6–12 (2007)
4. Boukerche, A., Oliveira, H.A.B.F., Nakamura, E.F., Loureiro, A.A.F.: Secure Localization Algorithms for Wireless Sensor Networks. *IEEE Communications Magazine* 46, 96–101 (2008)
5. Nagpal, R., Shrobe, H., Bachrach, J.: Organizing a global coordinate system from local information on an ad hoc sensor network. In: Zhao, F., Guibas, L.J. (eds.) *IPSN 2003*. LNCS, vol. 2634, pp. 333–348. Springer, Heidelberg (2003)
6. Niculescu, D., Nath, B.: Ad hoc positioning system (APS) using AoA. In: *Proceedings of IEEE INFOCOM 2003*, April 2003, vol. 3, pp. 1734–1743 (2003)
7. He, T., Huang, C., Blum, B.M., Stankovic, J.A., Abdelzaher, T.F.: Range-free localization and its impact on large scale sensor networks. *ACM Transactions on Embedded Computing Systems* 4, 877–906 (2005)
8. Niculescu, D., Nath, B.: DV based positioning in ad hoc networks. *Telecommunication Systems* 22, 267–280 (2003)
9. Thaeler, A., Ding, M., Cheng, X.: iTPS: an improved location discovery scheme for sensor networks with long-range beacons. *Journal of Parallel and Distributed Computing* 65, 98–106 (2005)

10. Liu, D., Ning, P., Du, W.K.: Attack-resistant location estimation in sensor networks. In: Proceedings of the 4th international symposium on Information Processing in Sensor Networks, April 2005, pp. 99–106 (2005)
11. Liu, D., Ning, P.: Improving key predistribution with deployment knowledge in static sensor networks. *ACM Transactions on Sensor Networks* 1, 204–239 (2005)
12. Liu, D., Ning, P.: Location-based pairwise key establishments for static sensor networks. In: Proceedings of ACM Workshop on Security in Ad Hoc and Sensor Networks 2003, October 2003, pp. 72–82 (2003)
13. Zhou, Y.: Security in wireless sensor networks and multicast. PhD. dissertation, University of Florida (August 2007)
14. Simic, S., Sastry, S.: Distributed localization in wireless ad hoc networks. Technical report UCB/ERL M02/26, UC Berkeley (2002)

A Methodology towards Usable Trust Management

Zheng Yan and Valtteri Niemi

Nokia Research Center, Nokia
zheng.z.yan@nokia.com, valtteri.niemi@nokia.com

Abstract. Trust management is emerging as a promising technology to facilitate collaboration among entities in an environment where traditional security paradigms cannot be enforced due to lack of centralized control and incomplete knowledge of the environment. However, prior art generally lack considerations on usable means to gather and disseminate information for effective trust evaluation, as well as provide trust information to users. This could cause a trust management solution to be hard to understand, use, and thus accept by the users. This paper proposes a user driven trust modeling and management method in order to design and develop a usable trust management solution that could be easily accepted by the users towards practical deployment. We illustrate how to apply this method into the design of a mobile application's reputation system in order to demonstrate its effectiveness.

1 Introduction

The concept of trust has been studied in disciplines ranging from economic to psychology, from sociology to medicine, and to information science. It is hard to say what trust exactly is because it is a multidimensional, multidisciplinary and multifaceted concept [1]. We can find various definitions of trust in the literature. Common to these definitions are the notions of confidence, belief, faith, hope, expectation, dependence, and reliance on the goodness, strength, reliability, integrity, ability, or character of a person or thing [2]. Generally, a trust relationship involves two parties: a trustor and a trustee. The trustor is the person or entity who holds confidence, belief, etc. on the reliability, integrity, ability, etc. of another person or thing, which is the object of trust - the trustee.

Although trust has been recognized as a complicated concept hard to narrow down, the critical characteristics of trust can be summarized. Trust is subjective because the level of trust considered sufficient is different for each individual in a certain situation. It is the subjective expectation of the trustor on the trustee that could influence the trustor's belief. Trust is also dynamic as it is affected by many factors. It can further develop and evolve due to good experiences about the trustee. It is sensitive to be decayed caused by bad experiences.

Recently, trust management has been emerging as a promising technology to facilitate collaboration among entities in a digital environment where traditional security paradigms cannot be enforced due to lack of centralized control and incomplete knowledge of the environment. Trust management is concerned with: collecting the information required to make a trust relationship decision; evaluating the criteria

related to the trust relationship as well as monitoring and re-evaluating existing trust relationships; and automating the process [3]. Various trust management systems have been described in the literature. One important category consists of reputation based trust management systems. Trust and reputation mechanisms have been proposed in various fields such as distributed computing, agent technology, GRID computing, and component software [4-7]. Recently, many mechanisms have been developed for supporting trusted communications and collaborations among computing nodes in a distributed system [8-11]. These mechanisms are based on digital modeling of trust for trust evaluation and management.

Due to the subjective characteristic of trust, trust management needs to take the trustor's criteria into consideration. For a digital system, it is essential for the user's device to understand the user's trust criteria in order to behave as her/his agent for trust management. Generally, it is not good to require a user to make a lot of trust related decisions because that would destroy usability. Also, the user may not be informed enough to make sound decisions. Thus, establishing trust is quite a complicated task with many optional actions to take. Trust should rather be managed automatically following a high level policy established by the trustor [12]. User-device interaction is needed if the device inquires the user's trust criteria in various contexts. This would require a friendly user interface to a) collect useful information for trust evaluation and management; b) present the evaluation results in a comprehensive manner to the user; and c) disseminate individual experiences to other devices as recommendations or contribute to reputation generation. There could also be other novel approaches that can help us to design a usable trust management system.

In this paper, we propose a user driven trust modeling and management method in order to design and develop a usable trust management solution that can be easily accepted by the users towards practical deployment. Our focus is to manage trust between a user and a digital system, which is either a device or a digital service or a software application consumed by the user.

The rest of the paper is organized as follows. Section 2 gives a brief overview of the literature background. Section 3 proposes the method of user driven trust modeling and management. In Section 4, we illustrate how to apply this method by taking the design and development of a mobile application's reputation system as an example. We further discuss the advantages of our method in Section 5. Finally, conclusions and future work are presented in the last section.

2 Background

There are two main categories of trust management study. One is psychological and sociological study on trust. The other is engineering study on trust in a computational way or for the purpose of trusted computing. We aim to study different methods applied in the literature in order to propose a feasible approach that can be adopted in practice towards usable trust management.

2.1 Psychological and Sociological Study on a Trustworthy System

Basso et al. examined the early formation of trust and the likelihood that a shopper will return to a website for subsequent purchases [13]. Two hypotheses were proposed

in the study and corresponding experiments were conducted to prove them. The first hypothesis is that the presence of voice and interactivity should each lead to higher ratings on trustworthiness and other positive attributions. The second hypothesis assumed that trust in a store's reliability and the UI's ability to engage the shopper should significantly predict purchase intent. Based on the hypothesis, the authors studied the shopper's behavior after first impression and after real experience based on initial trust. Results indicated that real-time interactivity, but not voice, increased judgments of friendliness and of the trustworthiness of the salesperson.

Lumsden and MacKay presented and discussed the results of a study which took an initial look at whether consumers with different personality types (a) are generally more trusting and (b) rely on different trust cues during their assessment of first impression vendor trustworthiness in B2C e-commerce [14]. They developed a questionnaire to serve as an initial investigation into the effect of personality type on consumers' trust and perception of importance of trust triggers. A five-point Likert scale was applied to let respondents respond their feedback of each questionnaire item. The applied research method is helpful to investigate the trust influencing factors and users' opinions on user-device interaction designs.

Herlocker, Konstan and Riedl studied explanation's influence on user's acceptance of ACF (Automated Collaborative Filtering) systems [17]. They addressed explanation interfaces for ACF systems – *how* they should be implemented and *why* they should be implemented. A model for explanations based on the user's conceptual model of the recommendation process was proposed. User experimental results demonstrated what components of an explanation are the most compelling. To address *why*, experimental evidence was presented to show that providing explanations can improve the acceptance of ACF systems. It has been proved that user experiment study greatly help in designing a trustworthy system user interface.

Pu and Chen used a qualitative survey to find research focus – explanation interface and the related design issues [18]. They further used pilot study and interview; post-survey discussion/interview; significant scale empirical study; paired samples t-test, and five-point Likert scale to conduct continuous research.

An integrated model of trust in electronic commerce was proposed in [19]. This model serves as the theoretical foundation to study the impact of trust on the success of electronic commerce. The model was developed by studying existing research in diverse fields such as psychology, social psychology, relationship theory, and human machine interaction, then integrated all valuable results into a comprehensive model. This method is beneficial for us in order to propose a new method built upon the advantages of previous work.

One promising approach of trust modeling aims to conceptualize trust based on user studies through a psychological or sociological approach (e.g. a measurement scale). This kind of research aims to prove the complicated relationships among trust and other multiple factors in different facets. Two typical examples are the initial trust model proposed by McKnight, Choudhury, and Kacmar [15] that explained and predicted customer's trust towards an e-vender in an e-commerce context, and the Technology Trust Formation Model (TTFM) studied by Li, Valacich, and Hess [16] to explain and predict people's trust towards a specific information system. For other examples, Gefen proved that familiarity builds trust [27]; Pennington, Wilcox, and Grover tested that one trust mechanism, vendor guarantees, has direct influence on system trust [28]; Bhattacharjee studied three key dimensions of trust: trustee's

ability, benevolence and integrity [29]; Pavlou and Gefen (2004) explained that institutional mechanisms engender buyer's trust in the community of online auction sellers [26]. This measurement scale based study could help us work out a trust construct. The result could instruct the design of a trust management system. Unfortunately, it is impossible to apply the graphic or linguistic trust constructs directly for digital trust evaluation and management.

The above work aims to conceptualize trust based on user studies through a psychological or sociological approach. The trust models generated based on this approach are generally linguistic or graphic [1]. They do not quantify trust for machine processing purposes. Therefore, the achieved results could only help people understanding trust more precisely. They generally work as guidelines or organizational policies for developing a trustworthy digital system or designing a trustworthy user interface. Although little work has been conducted to integrate psychological, sociological and computational theories together, we believe, however, that the psychological and sociological study could further play as a practical foundation of computational trust – modeling trust for a digital processing purpose.

2.2 Computational Trust

The method to specify, evaluate, set up and ensure trust relationships is referred to as a trust model [2]. Computational trust is a technical approach applied to represent trust for the purpose of trust calculation and digital processing. Regarding computational trust, we found quite a number of studies in the literature [1]. One of the earliest formalizations of trust in computing systems was done by Marsh in 1994 [20]. In his approach, he integrated the various facets of trust from the disciplines of economics, psychology, philosophy and sociology. Since then, many trust models have been constructed for various computing paradigms such as ubiquitous computing, peer-to-peer (P2P) networks, and multi-agent systems. For example, Abdul-Rahman and Hailes used discrete integer numbers to describe the degree of trust in virtual communities [22]. Then, simple mathematic, such as minimum, maximum, and weighted average, is used to calculate unknown trust values through concatenation and multi-path trust propagation. Buchegger and Le Boudec designed a distributed reputation system using a Bayesian approach for P2P and mobile ad-hoc networks, in which the second-hand reputation rating is accepted only when it is not incompatible with the primary rating [23]. In almost all of these studies, trust is accepted as a subjective notion by all researchers, which brings us to a problem: how to measure trust? Translation of this subjective concept into a machine readable language becomes a main objective. However, most of above studies focus on computational trust expression and calculation. Some factors or subjective policies used in the models were generally hidden in the system without any confirmation from the users if they were the trustors.

Sun, Yu, Han and Liu presented an information theoretic framework to quantitatively measure trust and model trust propagation in ad hoc networks [8]. In the proposed framework, trust is a measure of uncertainty with its value represented by entropy. The authors develop four axioms that address the basic understanding of trust and the rules for trust propagation. Based on these axioms two trust models are introduced: an entropy-based model and a probability-based model, both satisfy all the axioms. The only doubt of this work is whether the fundamental axioms are accepted by normal users, which could be an issue in practical deployment.

Xiong and Liu presented five trust parameters used in PeerTrust, namely, feedback a peer receives from other peers, the total number of transactions a peer performs, the credibility of the feedback sources, a transaction context factor, and a community context factor [5]. By formalizing these parameters, a general trust metric is presented. It combines these parameters in a coherent scheme. This model can be applied into a decentralized P2P environment. It is effective against dynamic personality of peers and malicious behaviors of peers. This work did not consider P2P system users' concern regarding all trust parameters and feedback distribution and collection. It applied a laboratory simulation to prove trust evaluation metric and its efficiency against malicious peers.

2.3 Applied Methods

The study of a trustworthy system is wide. We briefly summarize some methods applied in the related work in Table 1. Herein, we do not involve some interesting research due to its infancy. For example, a trust model could be derived based on a bio-inspired approach, such as an ant colony system [30].

Table 1. Research methods for establishing a trustworthy system

Examples	Research Methods
Basso, Goldberg, Greenspan and Weimer [13]	Hypothesis based initial study; trust model design based on experimental results on users
Lumsden and MacKay [14]	Questionnaire-based survey with five-point Likert measurement scale
McKnight, Choudhury, and Kacmar [15]; Li, Valacich and Hess [16]; Gefen [27]; Pennington, Wilcox, and Grover [28]; Bhattacharjee [29]; Pavlou and Gefen [30]	A number of measurement scales developed to study trust constructs and trust relationships with other factors
Herlocker, Konstan and Riedl [17]	Prove research hypothesis through user experimental study
Pu and Chen [18]	Qualitative survey; pilot study and interview; post-survey discussion/interview; significant scale empirical study; paired samples t-test, and five-point Likert scale
Kini and Choobineh [19]	Integration of previous research results
Abdul-Rahman and Hailes [22]	A discrete trust model of virtual communities, which is based on experience and reputation. An example application was illustrated
Buchegger and Le Boudec [23]	A continuous trust model based on a modified Bayesian estimation approach. Simulation proof on its performance
Sun, Yu, Han and Liu [8]	Trust modeling and evaluation based on axioms with laboratory simulation proof
Xiong and Liu [5]	Laboratory simulation proof on the proposed trust metric

Obviously, a thorough understanding of both the psychological/sociological and engineering aspects of trust is necessary in order to develop a usable trust management solution. However, the psychological and sociological trust study lacks a way towards digital trust management, while engineering study lacks a basic sociological and psychological foundation in order to convince normal users for easy acceptance. Current computational trust study generally lacks sociological and psychological support. Therefore, it is hard to predict if a trust management system built upon it could be easily accepted and widely used. A gap exists between these two categories of trust research. The reason could be they are solving different research issues. But for developing a practical trust management system, we need to apply the advances of both researches and make computational trust derived from social trust finally benefit the users.

3 A Method of User Driven Trust Modeling and Management

To overcome the above gap, we propose a user driven trust modeling and management method. We aim to design and develop a usable trust management solution that can be easily accepted by the users towards practical deployment. Our focus is to support user-device and user-system trust. For low level trust management (e.g. trustworthy network routing) without any concern and involvement of users (in the areas such as ad hoc networks and wireless sensor networks), this method may not be applicable since it only treats the cases with the user as the trustor. Herein, the term “user-driven” means that user study is applied in every step of our research in order to prove users’ acceptance of our system design and development. A user-driven computational trust model achieved through applying this method will play as a core of the trust management system. This trust model is different from traditional trust models in e-commerce [15, 16]. This lies in the fact that the model is achieved by formalizing an empirical trust construct in a mathematical way. It reflects the users’ perspectives and integrates the advantages of computational trust and social trust studies. Additional user experimental studies will be further conducted in order to provide a trustworthy human-device interaction design required in the trust management system. Figure 1 presents our research method with four steps.

Step 1 aims to figure out trust constructs for computational trust modeling. Firstly, we propose a number of hypotheses based on literature theory and knowledge. We then design a measurement scale to conduct user experiments on a suitable number of users. We further apply a psychometric method to analyze the experimental data in order to find out the constructs and sub-constructs of trust. The above procedure could be repeated in order to achieve a stable trust construct. For example, the user experiments should be conducted a couple of times in order to extract principle factors of trust construct, optimize the measurement scale and study the causal relationships among those factors [15]. This is the psychological and sociological study of trust model. The result is a clear construct of trust based on experimental data collected. This step also answers the question: what data or information should be collected in order to do trust/reputation evaluation.

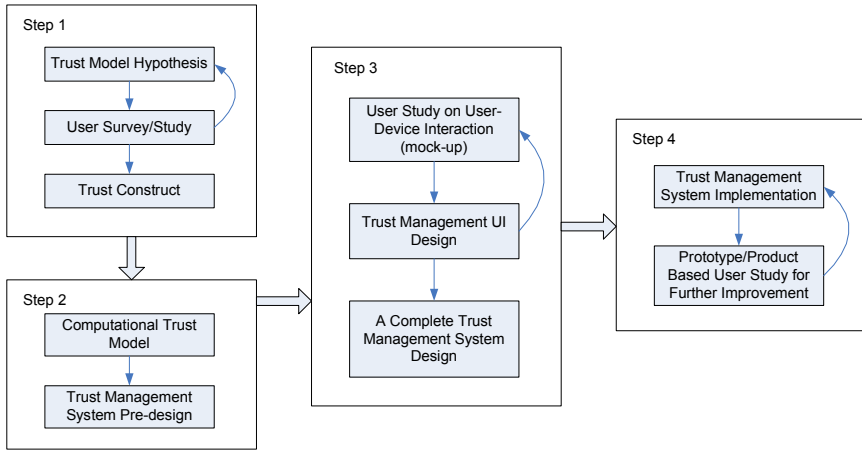


Fig. 1. A procedure of user driven trust modeling and management

Step 2 aims to work out a user driven trust model and the pre-design of trust management system. In the way towards digital management of trust, we should further work out a computational trust model on the basis of the trust construct achieved in the first step. The computational trust model should reflect the principle factors of trust construct and their causal relationships. Laboratory simulation based proof is essential since the user study itself may not help to overcome a number of malicious behaviors or serious attacks. The computational trust model proposed based on the user study should be further improved and optimized according to the simulation results. We call what we achieve as a user driven trust model. At this time, the trust management system can be pre-designed according to the achieved model.

In Step 3, we conduct relevant user study about the pre-designed trust management system. User's feedback will be collected and analyzed. The user experiment could be mockup based and repeated several times in order to improve/optimize and pre-prove the user-device interaction design for trust management. After this, a complete trust management system design (i.e. both backend design and front end user interface design) can be achieved.

In Step 4, a prototype or a trial product is implemented. Real system usage experiences and feedbacks are collected from the users for further improvement. The improved system could be further evaluated for additional optimization.

It is important to note that some sub-steps listed above are iterative in order to achieve either a good model or a usable design. The purpose is to consider the users' preference and expectation as early as possible thus effectively save the cost of the system development and greatly enhance the users' acceptance.

4 Applicability Study

In this section, we take the design of a mobile application reputation system as an example to illustrate how to apply the proposed method into practice. This project has been conducted for two years. A mobile application is a software package that can be

installed and executed in a mobile device, for example, a mobile email client that can help a mobile user to check and manage his/her email using a mobile phone. Generally, the mobile applications developed by various vendors can be downloaded from a web site or received from another device for installation. The trustworthiness of a mobile application influences the user's purchase and usage and thus becomes a crucial issue that impacts its final success.

We aim to design and develop a common and usable reputation system for various mobile applications that could help the mobile users' purchase and usage. We hope to achieve sound usability; otherwise the users could not accept the system. This means user-device interaction for trust management should be designed in a usable way. Thus, the system can be easily followed and accepted by mobile users. It is ideal that the users feel very natural and normal with regard to data extraction and dissemination for trust/reputation evaluation and trust/reputation information presentation.

4.1 Trust Construct Analysis

In order to collect users' usage experiences in an easy and usable way for trust and reputation evaluation, we proposed a hypothesis: a user's trust in a mobile application can be reflected through his/her usage behaviour [21]. Then, we designed a questionnaire with seven-point Likert measurement scale to conduct a user survey. We hypothesize several types of usage behaviors that reflect a user's trust: Using Behavior (UB) related to normal usage, Reflection Behavior (RB) about application performance reaction and Correlation Behavior (CB) with regard to the usage difference on applications with similar functions. All types of behaviors comprise the user's trust behavior (i.e. a trustor's actions to depend on, or make her/him vulnerable to a trustee) related to a mobile application.

Firstly, we conducted a pre-experiment with more than 300 participants and applied Exploratory Factor Analysis (i.e. Principle Component Analysis) in order to optimize the questionnaire [21]. Then, we ran a formal experiment with more than 1500 participants to figure out a rational trust behaviour construct through Principle Component Analysis, correlation analysis, reliability analysis and Confirmatory Factor Analysis. Based on the results achieved, we got the main factors and construct of trust behavior that contribute to the calculation of the user's trust in a mobile application. Concretely, the PCA, CFA, reliability and correlation analysis showed that far from being unitary, the trust behavior has multiple dimensions. We explored and proved three significant dimensions: the using behavior, the reflection behavior, and the correlation behavior; and figured out their internal constructs and external variables (e.g. personality, brand, experienced quality and personal motivation) that may influence them [21, 24]. Those trust behaviors can be automatically monitored and thus recorded when the mobile user is using the application. Therefore, the mobile device can automatically collect useful information for trust/reputation evaluation in a natural and usable way.

4.2 Computational Trust Model and Trust Management Pre-design

A computational trust model can be further proposed based on the trust construct achieved from Step 1. It expresses the principle factors related to trust and their

relationships in a mathematical method. This model was achieved by formalizing the trust behavior constructs. We reported this work in another paper in details. Briefly, a general trust metric of mobile application i can be expressed as:

$$T(i) = T(i)_o + \alpha T(i)_{UB} + \beta T(i)_{RB} + \gamma T(i)_{CB} \quad (1)$$

where α, β, γ denote the normalized weight factors for using behavior evaluation $T(i)_{UB}$, reflection behavior evaluation $T(i)_{RB}$, and correlation behavior evaluation $T(i)_{CB}$. $T(i)_o$ stands for an original trust value that may be influenced by some external variables, such as personality, brand, personal motivation and experienced device quality. $T(i)_o$ could also be a past trust value generated according to the past trust behaviors.

We further conducted laboratory simulations to optimize and improve the computational model. Target is to evaluate the formalization with example usage models. If necessary, we should further examine its robustness against various malicious behaviors and attacks. This model can be used to calculate each user's direct trust in a mobile application. It can also be applied to achieve a reputation value of the mobile application based on a number of users' trust behaviors. On the basis of the above achieved model, a reputation trust management system can be pre-designed.

As shown in Figure 2, a distributed client-server system structure is adopted. The client package can be installed at a number of mobile devices ($MD_i, i = 1, \dots, n$). It contains a user behavior monitor which monitors trust behaviors and inputs statistical data into a secure storage (Trust Data), which could be located inside the device platform and has a secure channel to communicate with the usage behavior monitor and the trust/reputation information presenter. The statistical data can be accessed by a data interpreter for a) trust evaluation on the user's trust status regarding a specific application according to the computational trust model; b) data dissemination to share local trust information with a reputation service provider; c) reputation extraction to receive a mobile application's reputation information from a reputation service provider. The data interpreter is a secure mechanism to access the user's usage statistical data from the secure storage (Trust Data) since these data are private information. We design the data interpreter based on the trusted computing technology [31]. It is the only authorized mechanism to access and unseal the protected usage information. The reputation extraction can be tailored based on the mobile user's preference, either with a public reputation extraction policy or a group reputation extraction policy (e.g. a social networking based reputation). In addition, a trust/reputation information presenter is applied to show trust/reputation information (either an indicated trust or reputation value or detailed information about how this value is achieved and certified) to the user in order to aid his/her usage of the mobile application.

At the reputation service provider side, a trust value receiver receives the trust value of the mobile application automatically or by request from a number of mobile devices ($MD_i, i = 1, \dots, n$). A reputation generator tries to create a reputation value for each mobile application. Herein, the reputation could be generated based on public (a number of users) usage statistics. But due to privacy concern, we apply another algorithm that aggregates the trust values (calculated at each mobile device according to the computational trust model) together following a number of policies in order to overcome malicious attacks. The reputation information for each mobile application is

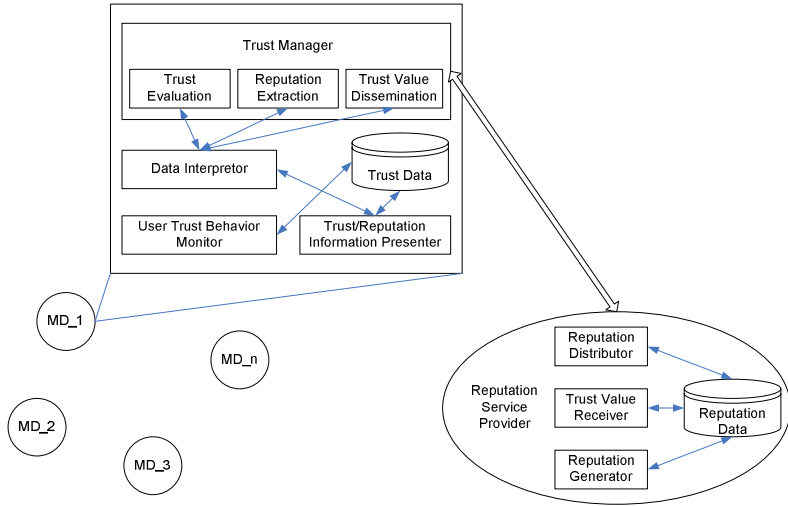


Fig. 2. Structure of a mobile application reputation system

saved in a secure storage (Reputation Data). This information can be retrieved and distributed to a number of mobile devices ($MD_i, i = 1, \dots, n$) through a reputation distributor. The reputation distributor receives or motivates a reputation retrieve request and provides the expected reputation value to the requestor.

4.3 User Study on Pre-designed Trust Management System

At this point, we were clear about what user-device interactions are needed in the underlying system pre-design. Clearly, data extraction for trust management can be conducted automatically with few user interactions (e.g. allowing the device to share the direct trust information anonymously). What we need to study is why, how, what and when to show the trust/reputation information to the user and the corresponding design for trustworthy user-device interaction (e.g. how to make user feel convenient to share his personal usage experiences and trust information). A practical strategy could be that the user is inquired to agree sharing his/her personal trust information in order to retrieve the reputation information of a mobile application.

For each required user-device interaction in the pre-designed system, we should conduct corresponding user study in order to design an easily accepted user interface with the users’ considerations kept in mind. In this case, we need to study the following:

- a) Whether it is helpful to provide trust or reputation information to the users when they are using a mobile application. This study aims to solve the issue why interaction between user and device is needed.
- b) How to display the trust/reputation information and what contents should be provided to the users and in which way. This study aims to solve the issues how to interact and what to be interacted.

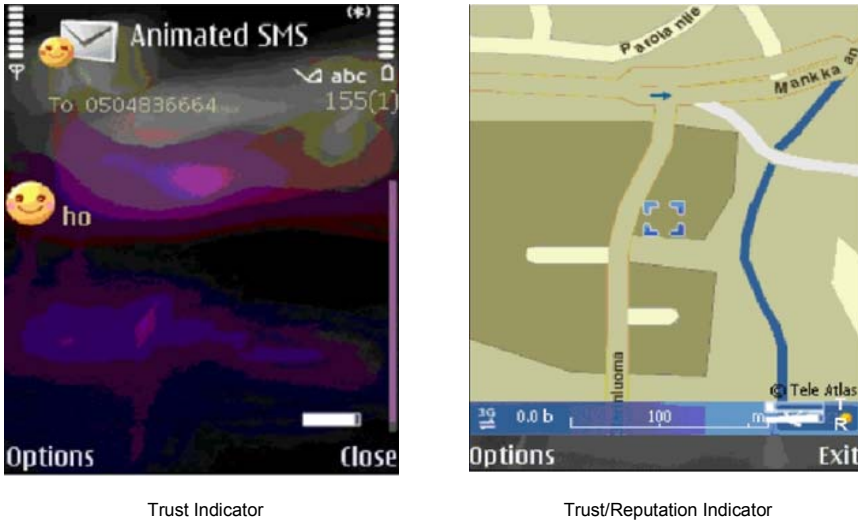


Fig. 3. Trust indicator and trust/reputation indicator

- c) At which moment is user-device interaction required, (e.g. whether a user's confirmation is needed for sharing his usage history with others or a reputation service provider). This study aims to answer the question when interaction is required.

We conducted a mock up based user study to explore a) in both China and Finland with about 90 participants, respectively. The results of an analysis of variance and paired samples t tests showed that it has statistical significance to indicate the trust value or the trust/reputation values of a mobile application during its usage in both countries. In the trust indicator test, we found significant main effects of the trust value [$F(2, 50) = 17.651, p < 0.001$ in China; $F(2, 56) = 1035.187, p < 0.001$ in Finland], indicating that the willingness of continuously using a mobile application increased from the trust value low to high. In the trust/reputation indicator test, we found significant main effects of the indicated trust value [$F(2, 54) = 42.148, p < 0.001$ in China; $F(2, 48) = 707.860, p < 0.000$ in Finland], and the indicated reputation value [$F(2, 54) = 28.734, p < 0.001$ in China; $F(2, 48) = 1009.887, p < 0.000$ in Finland], indicating that the willingness of continuous usage increased from the indicated trust value low to high and from the indicated reputation value low to high.

For studying b) and c), we interviewed about 120 participants in both China and Finland to get their feedback about user interface design and trust/reputation indicator design. Based on the interview, the mock-up designed trust indicator and trust/reputation indicator are preferred and accepted by the participants, as shown in Figure 3. But some participants prefer that the indicator should be shown transparently at the usage beginning and then decaying gradually. We understood that it is applicable to provide a personalized trust information display solution in order to satisfy different users' preference. For the users who would like to keep their usage privacy, we could only show the trust indicator since the direct trust value can be calculated by

their personal devices. Regarding c), most of participants think the usage information is private in both countries (73.3% in Finland and 90% in China), but most of them would like to share this information with privacy protection (70% in Finland and 60% in China). We found the necessity to apply suitable technologies to allow the users to control their usage information sharing with anonymous support. Regarding the detailed trust/reputation information (such as how the trust/reputation value is calculated and who certifies it), it is not conveniently to show it together with the indicator since it will ruin the usability. People would like to check it when, for example, trust value and reputation value are quite different in Finland or the application scenario is very important in both Finland and China. In both countries, about half participants think menu is their preferred way to get the trust information details (50% in Finland and 60% in China). More participants like touching the indicator to access the detailed trust information in Finland (40%) than in China (3.3%). In both countries, some people prefer short-cut key access; the percentage is a bit higher in China (26.7%) than in Finland (10%).

According to the above user study results, the system user interface design can be worked out to satisfy users' preferences and considerations. Based on the guidelines summarized from our user studies, we then worked out a complete reputation system design for implementation.

4.4 Prototype Based User Study for Further Improvement

A prototype/product trial could now be implemented on the basis of the user driven trust modeling and management system design. Based on the prototype or trial system, real system usage experiences and feedback can be collected from the users through survey, field study or focus group interview for further system improvement.

5 Discussions

We believe our method hold a number of advantages over existing methods. Firstly, the user driven trust model is proposed based on a wide user survey. Statistical data analysis and Structural Equation Modeling plays as the foundation that could help us generate a trust model easily accepted by the users [25]. Thus our method overcomes the weakness of current computational trust models that were built beyond any proof of users.

Secondly, the computational trust model is proposed on the basis of trust constructs achieved from the user study. It is further improved and optimized based on laboratory experiments. This compensates the problem of linguistic or graphic trust models generated purely from the user study and its hardness to be directly applied into trust management for a digital system.

Thirdly, sound usability could be easily achieved if applying our method. Based on the pre-design of the user driven trust management system, we can study why, when, how and what should be interacted between the users and their devices. These user study results play as the design guidelines for developing a usable trust management system. In addition, the method itself could study the applicability of

our idea that aims to improve system usability and release the burden of user-device interaction in itself.

On the other hand, we also got some lessons from our research and experiments. Firstly, we realized that it is important to seriously study the theories behind our hypothesis before conducting the trust construct study based on the user survey. However, the theoretical support was hard to find since the mobile application is a new area (unlike e-commerce on-line trust). Actually, trust behavior was rarely studied in the literature. The Exploratory Factor Analysis can indicate the problems of our questionnaire, but not enough to improve and optimize it (e.g. adding new items). We think an additional survey could be needed in order to achieve a more stable measure than the current one. Secondly, the user driven trust model is formalized directly based on the trust construct. It could be incomprehensive and imprecise, unlike many existing computational trust models. How to overcome this weakness is a challenge of this methodology. Or we have to make some trade-off in the trust modeling in order to achieve usability. Finally, we found a special privacy concern during the user studies on user-device interaction. We recognized that trust and privacy could conflict with each other. How to enhance privacy and achieve trust management is a practical challenge that motivates our further research.

6 Conclusions and Future Work

In this paper, we presented our motivations for developing a usable trust management solution. We briefly overviewed the existing research methods applied for establishing a trustworthy system. We found that it lacks a practical approach that could help us design and develop a usable trust management system. Furthermore, we proposed a method called user driven trust modeling and management to overcome the weakness of the existing methods. We illustrated its applicability through applying it into the design and development of a reputation system for mobile applications. This paper contributes on two folds. Firstly, it motivated to drive trust modeling and management from the users' points of view towards practical system deployment. Secondly, it proposed an applicable and cross-disciplinary method to design and develop a usable trust management system through integrating the advances of both psychological/sociological trust study and computational trust study.

At present, we are developing a reputation system for mobile application based on the achieved trust management system design driven by a series of user studies. Further proof and improvement of our method could be based on the feedback collected from prototype users. Regarding the methodology proposed in this paper, we will attempt to apply it into other applications, such as various mobile internet services.

Acknowledgement

The authors would like to thank Dr. Yan Dong, Dr. Conghui Liu, Prof. Rong Yan and Prof. Guoliang Yu's cooperation and contribution in the project related to this paper work.

References

- [1] Yan, Z., Holtmanns, S.: Computer Security, Privacy and Politics: Current Issues, Challenges and Solutions. In: Trust Modeling and Management: from Social Trust to Digital Trust. IGI Global (2008)
- [2] Yan, Z.: Trust Management for Mobile Computing Platforms. Doctoral dissertation, Dept. of Electrical and Communications Engineering, Helsinki Univ. of Technology (2007)
- [3] Grandison, T., Sloman, M.: A Survey of Trust in Internet Applications. *IEEE Communications and Survey* 3(4), 2–16 (2000)
- [4] Resnick, P., Zeckhauser, R.: Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay's Reputation System. In: Baye, M. (ed.) *Advances in Applied Microeconomics: The Economics of the Internet and E-Commerce*, vol. 11, pp. 127–157. Elsevier, Amsterdam (2002)
- [5] Xiong, L., Liu, L.: PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. *IEEE Transactions on Knowledge and Data Engineering* 16(7), 843–857 (2004)
- [6] Herrmann, P.: Trust-Based Procurement Support for Software Components. In: *Proceedings of the 4th International Conference of Electronic Commerce Research (ICECR 2004)*, pp. 505–514 (2001)
- [7] Walsh, K., Sirer, E.G.: Fighting Peer-to-Peer SPAM and Decoys with Object Reputation. In: *Proceedings of the Third Workshop on the Economics of Peer-to-Peer Systems (P2PECON)*, August 2005, pp. 138–143 (2005)
- [8] Sun, Y., Yu, W., Han, Z., Liu, K.J.R.: Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks. *IEEE Journal on Selected Area in Communications* 24(2), 305–317 (2006)
- [9] Zhang, Z., Wang, X., Wang, Y.: A P2P Global Trust Model Based on Recommendation. In: *Proceedings of 2005 International Conf. on Machine Learning and Cybernetics*, vol. 7, pp. 3975–3980 (2005)
- [10] Theodorakopoulos, G., Baras, J.S.: On Trust Models and Trust Evaluation Metrics for Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications* 24(2), 318–328 (2006)
- [11] Lin, C., Varadharajan, V., Wang, Y., Pruthi, V.: Enhancing Grid Security with Trust Management. In: *Proceedings of IEEE International Conf. on Services Computing*, pp. 303–310 (2004)
- [12] Yan, Z., Prehofer, C.: An Adaptive Trust Control Model for a Trustworthy Component Software Platform. In: Xiao, B., Yang, L.T., Ma, J., Muller-Schloer, C., Hua, Y. (eds.) *ATC 2007*. LNCS, vol. 4610, pp. 226–238. Springer, Heidelberg (2007)
- [13] Basso, A., Goldberg, D., Greenspan, S., Weimer, D.: Emotional and Cognitive factors Underlying Judgments of Trust E-Commerce. In: *Proceedings of the 3rd ACM conference on Electronic Commerce, EC 2001* (October 2001)
- [14] Lumsden, J., MacKay, L.: How Does Personality Affect Trust in B2C E-Commerce? In: *Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet, ICEC 2006* (August 2006)
- [15] McKnight, D.H., Choudhury, V., Kacmar, C.: Developing and Validating Trust Measures for E-Commerce: an Integrative Typology. *Information Systems Research* 13(3), 334–359 (2002)

- [16] Li, X., Valacich, J.S., Hess, T.J.: Predicting User Trust in Information Systems: a Comparison of Competing Trust Models. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences, January 2004, p. 10 (2004)
- [17] Herlocker, J.L., Konstan, J.A., Riedl, J.: Explaining Collaborative Filtering Recommendations. In: Proceedings of the 2000 ACM conference on Computer supported cooperative work, CSCW 2000 (December 2000)
- [18] Pu, P., Chen, L.: Trust Building with Explanation Interfaces. In: Proceedings of the 11th international conference on intelligent user interfaces, IUI 2006 (January 2006)
- [19] Kini, A., Choobineh, J.: Trust in Electronic Commerce: Definition and Theoretical Considerations. In: Proceedings of the 31st Hawaii International Conference on System Science, January 1998, vol. 4, pp. 51–61 (1998)
- [20] Marsh, S.: Formalising Trust as a Computational Concept. Doctoral dissertation, University of Stirling (1994)
- [21] Yan, Z., Niemi, V., Dong, Y., Yu, G.: A User Behavior Based Trust Model for Mobile Applications. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) ATC 2008. LNCS, vol. 5060, pp. 455–469. Springer, Heidelberg (2008)
- [22] Abdul-Rahman, A., Hailes, S.: Supporting Trust in Virtual Communities. In: Proceedings of the 33rd Hawaii International Conference on System Sciences (2000)
- [23] Buchegger, S., Le Boudec, J.Y.: A Robust Reputation System for P2P and Mobile Ad-Hoc Networks. In: Proceedings of the 2nd Workshop Economics of Peer-to-Peer Systems (2004)
- [24] Dong, Y., Yan, R.: Technical report: Trust Behaviors of Mobile Application Usages (October 2008)
- [25] MacCallum, R.C., Austin, J.T.: Applications of Structural Equation Modeling in Psychological Research. *Annual Review of Psychology* 51, 201–226 (2000)
- [26] Pavlou, P., Gefen, D.: Building effective online marketplaces with institution-based trust. *Information Systems Research* 15(1), 37–59 (2004)
- [27] Gefen, D.: e-Commerce: The Role of Familiarity and Trust. *Omega: Internet. J. Management Sci.* 28(6), 725–737 (2000)
- [28] Pennington, R., Wilcox, H.D., Grover, V.: The Role of System Trust in Business-to-Consumer Transactions. *Journal of Management Information Systems* 20(3), 197–226 (2004)
- [29] Bhattacharjee, A.: Individual Trust in Online Firms: Scale Development and Initial Test. *Journal of Management Information Systems* 19(1), 211–241 (2002)
- [30] Yan, L.: Understanding Network Mobility in Pervasive Markets: Realistic Human Shopping Behavioral Model. In: Lytras, M.D., Carroll, J.M., Damiani, E., Tennyson, R.D. (eds.) WSKS 2008. LNCS, vol. 5288, pp. 114–122. Springer, Heidelberg (2008)
- [31] TCG TPM Specification v1.2 (2003),
<https://www.trustedcomputinggroup.org/specs/TPM/>

Formalizing Trust Based on Usage Behaviours for Mobile Applications

Zheng Yan¹ and Rong Yan²

¹ Nokia Research Center, Helsinki, Finland

zheng.z.yan@nokia.com

² Beijing International Studies University, Beijing, China

rongyanallen@tom.com

Abstract. A mobile application is a software package that can be installed and executed on a mobile device. Which mobile application is more trustworthy for a user to purchase, download, install, consume or recommend becomes a crucial issue that impacts its final success. This paper proposes a computational trust model based on users' behaviors, which assists the evaluation and management of the mobile application's trust with user friendliness. We achieve our model through formalizing a trust behavior construct achieved from a user survey experiment though principle component analysis, reliability analysis, correlation analysis and confirmatory factor analysis. It is indicated that a user's trust behavior is a multidimensional construct composed of three main aspects: using behavior, reflection behavior, and correlation behavior. Particularly, we simulate a set of test data to visualize the validity of the formalization.

1 Introduction

A mobile application is a software package that can be installed and executed on a mobile device (e.g. a mobile phone), for example, a mobile email client to access emails. Generally, this software package developed by various vendors can be downloaded for installation. Which mobile application is more trustworthy for a user to purchase, download, install, consume or recommend becomes a crucial issue that impacts its final success.

A user's trust in a mobile application is, being highly subjective, inherently hard to measure. It is built up over time and changes with the use of the application due to the influence of many factors. As it is an internal 'state' of the user, there is no way of measuring it directly. Fully supporting trust evaluation and management on mobile applications requires a number of usability studies regarding user's trust criteria/standards extraction, user's experience or feedback dissemination and user's trust decision reception. All of these may influence user friendliness of the mobile device.

Trust is important because it helps consumers overcome perceptions of uncertainty and risk and engages in "trust-related behaviors" (in short trust behaviors, i.e. a trustor's actions to depend on, or make her/him vulnerable to a trustee, e.g. provide the vendor personal information, engage in a purchase transaction, or act on vendor information such as financial advice) [10]. Credible information is gleaned after involved parties

have interacted for some time. Marsh reasoned that it might prove more suitable to model trust behavior rather than trust itself, removing the need to adhere to specific definitions [1]. In mobile application usage, we posit that credible information is gained only after a mobile device user has both engaged in trust behaviors (e.g., acting on using a mobile application) and assessed the trustworthiness of the application by observing the consequences of its performance, depending on it in his/her routine life.

This paper attempts to develop a trust model based on the trust behavior of mobile application usage. Thus, through auto-monitoring users' behaviors via user-device interactions, we can extract useful information for evaluating and managing trust of mobile applications in an autonomic and user-friendly measure. With this way, it is also possible to avoid heavy interactions that may be required by some existing trust management solutions, e.g. [2]. Developing such a trust model is significant for a mobile device to provide trust information to its user in order to leverage usage decision. It also benefits a mobile application provider that could provide suggestions to users in order to help them selecting a valuable mobile application.

The rest of the paper is organized as follows. Section 2 gives a brief overview of the literature. Section 3 introduces user experiment results – a construct of trust behavior. Section 4 designs a computational model that formalizes the above results in order to calculate trust value inside a mobile device. In section 5, we evaluate our model based on a set of simulated data. Finally, conclusions and future work are presented in the last section.

2 Background

2.1 Trust Model (From a Psychological View towards an Engineering View)

Current trust models have been developed based on specific security issues and also solely on knowledge, experience, practices, and performance history [3]. Much of the prior research in trust of automation has focused primarily on the psychological aspect [4]. But prior research lacks an integral understanding of both the psychological and engineering aspects of trust, which is essential for developing an appropriate trust model towards a trustworthy system that is easily accepted by the users.

We can find many proposals presented to link some of the psychological aspects of trust with engineering issues: “attempts have been made to map psychological aspects of trust (e.g. reliability, dependability, and integrity) to human-machine trust clusters associated with engineering trust issues such as reliability and security” [6]. Lance, et al. studied trust based on a number of influencing factors from the engineering and psychological points of view and tried to combine these factors in order to provide a comprehensive model [6]. Most of existing work follows the research steps that, what is trust referent, what are factors or aspects related to trust, and evaluate or assess trust based on those factors and aspects and try to manage trust accordingly [7]. But it is actually hard to computationally model some influencing factors, such as usability and a user's subjective factors [7]. Since trust is a subjective concept, assessing trust needs to understand the trustor's trust criteria regarding each factor or aspect, even for different contexts. This may raise a lot of interaction requirements between the user and device, and thus cause a usability issue that needs more efforts to overcome.

Initial trust refers to trust in an unfamiliar trustee, a relationship in which the actors do not yet have credible, meaningful information about, or affective bonds with, each other [9]. McKnight et al. proposed and validated measures for a multidisciplinary and multidimensional model of initial trust in e-commerce [10]. The model includes four high-level constructs: disposition to trust, institution-based trust, trusting beliefs, and trusting intentions, which are further delineated into sixteen measurable, literature-grounded sub-constructs. The cross-disciplinary nature of the trust typology in this study highlights the multiple, interrelated dimensions of e-commerce trust. The technology trust formation model (TTFM) is a comprehensive model of initial trust formation used to explain and predict people's trust towards a specific information system [11]. The above two models used the framework of the theory of reasoned action (TRA) to explain how people form initial trust in an unfamiliar entity [5]. Since the objective of TTFM model was to predict initial trust (trusting intention) before any actual interaction with the trustee, trust-related behavior was excluded from this model. McKnight et al. did not study trust behavior either, although they suggested conducting a study in which the ultimate outcome of interest, trust behavior, is directly measured in order to overcome the limitations of their study [10]. The results of the above studies could instruct the design of a trust management system. Unfortunately, it is impossible to apply the graphic or linguistic trust constructs directly for digital trust evaluation and management.

On the other hand, *short-term trust* is built up over the first interactions with a system and *long-term trust* is developed with the continuous use of a system over a longer period of time. *On-going trust* appeared in [10] concerns both the short-term trust and the long-term trust. In particular, the on-going trust could contribute to the trustee's reputation and thus greatly help other entities building up their initial trust. In our study, we mainly focus on the on-going trust evaluation based on the user's behaviors for individual direct trust evaluation. Based on the above literature, a user could hold a level of initial trust when he/she starts using a mobile application.

2.2 Trust Behavior

In TRA theory, it posits that beliefs lead to attitudes, which lead to behavioral intentions, which lead to the behavior itself [5]. Applying this theory, we propose that trusting beliefs (perceptions of specific mobile application attributes) lead to trusting intentions (intention to engage in trust behaviors of using a mobile application through user-device interaction), which in turn result in trust behaviors (using the application in various context). Additionally, numerous researchers have conceptualized trust as a behavior. Prove has been done in work collaboration and social communications [12-14]. Prior research has also confirmed a strong correlation between behavioral intentions and actual behavior, especially for software system usage [15, 16].

Muir is one of the first researchers to look at a decision process between supervisors and automated systems. She verifies the hypothesis that the supervisor's intervention behavior is based upon his/her trust in automation [4, 17]. She found a positive correlation between trust and use. The relationship between trust and interaction behavior is obvious. Lee and Moray [19] found that trust in a system partially explained system use, but other factors (such as the user's own ability to provide manual control) also influenced the system use. Although these studies have provided support

for Muir's theory, additional research is required to evaluate her hypotheses in more depth, especially in other domains. All above work plays as the foundation of our study: a user's trust in a mobile application can be evaluated based on the user's trust behaviors, which mostly can be monitored through user - device interaction.

Recently, trust management is emerging as a promising technology to facilitate collaboration among entities in an environment where traditional security paradigms cannot be enforced due to lack of centralized control and incomplete knowledge of the environment. However, prior arts generally lack considerations on the means to gather experiential evidence for effective trust evaluation. Many systems rely on a user to provide feedback [18]. Sometimes, it may not be appropriate or convenient to require him/her to provide feedback, especially for a mobile user. Another issue is different users may apply different scales in the feedback, which may cause confusion, even attacks. This introduces a requirement to largely automate the experiential evidence in a uniformed norm. Our work aims to overcome the above problems and support automatic evidence collection for trust evaluation and management.

3 A Trust Behaviour Construct

Our research question is what usage behaviors are related to the user's trust in a mobile application. We hypothesize several usage behaviors that reflect user's trust: Using Behavior (UB), Reflection Behavior (RB) and Correlation Behavior (CB), refer to Table 1. All behaviors comprise the user's trust behavior in a mobile application. They contribute to the calculation of the device's confidence on the user's trust in the mobile application. We applied a psychometric method to examine our hypotheses. We designed a questionnaire, taking Short Message Service (SMS) as a concrete example of mobile application. Each item in the questionnaire is a statement for which

Table 1. Hypotheses

BEHAVIOR TYPE	HYPOTHESES
§1 Using Behavior (UB)	§1.1 The user trusts a mobile application more, if he/she uses it more with more elapsed time, number and frequency of usages; §1.2 Trust in a mobile application could influence the user's behavior regarding risky, urgent or important tasks; §1.3 The user becomes more proficient in using a mobile application if he/she has experienced more features of the mobile application.
§2 Reflection Behavior (RB) (behaviors after confronting application problems or having good/bad experiences)	§2.1 Good/bad performance of a mobile application could increase/decrease the user's usage trust; §2.2 Good/bad application performance or usage experience could influence the user's behavior related to risky, urgent or important tasks.
§3 Correlation Behavior (CB) (behaviors correlated to similar functioned applications)	§3.1 For two similar functioned applications, higher usage rate (i.e. usage time, the number of usages and usage frequency) of one application means more trust in it; §3.2 For two similar functioned applications, the user would like to use more trusted one to do risky, urgent or important tasks; §3.3 Trust in a mobile application influences the behaviors of recommendation.

the participants need to indicate their level of agreement. The questionnaire is anchored using a seven-point Likert scale ranging from “strongly disagree” to “strongly agree”. Firstly, a pre-experiment with 318 participants was conducted in order to optimize our questionnaire [20]. Further, we ran a formal experiment with more than 1500 participants to study a trust behavior construct for mobile applications.

3.1 Scale Development

There are four basic parts in the scale. For the using behavior (UB), we designed a list of items about a) UB1: normal usage behavior; b) UB2: usage behavior related to context; and c) UB3: new feature related usage behavior. Regarding the reflection behavior (RB), we designed a number of items about a) RB1: bad performance reflection behavior; b) RB2: bad performance reflection behavior related to context; c) RB3: good performance reflection behavior; d) RB4: good performance reflection behavior related to context; e) RB5: bad experience reflection to context; and f) RB6: good experience reflection to context. In the part about the correlation behavior (CB), we design items about a) CB1: comparison of normal usage behavior; b) CB2: comparison related to context; and c) CB3: recommendation behavior. Finally, we designed a number of items in order to do external nomological validation. We attempt to study the following four external variables' influence on the user's trust behavior: a) personal motivation (PM); b) brand impact (BI); c) perceived device quality (DQ); and d) personality (P). The items designed for this part of test (especially for PM and DQ) were adapted based on the measurement of McKnight et al. [10].

3.2 Data Collection

The participants were chosen from three Chinese universities. 1575 subjects participated; 1120 responses (71.1%) were valuable and usable based on three selection criteria: 1) no empty questionnaire item response; 2) no regular pattern can be found from the responses; 3) the responses on all items are not the same (i.e. serious response). Among the selected subjects with valid responses, 671 (59.9%) were women and 449 (40.1%) were men; 43 participants were below 18 years and others were between 19-35 years. 502 (44.8%) participants major in science or technology, while 480 (42.9%) in arts. Except one sample information is missing, the rest major in integration of science and art. According to the survey, 419 (37.4%) participants had experiences of using the Internet accessed applications (e.g. a web browser), 864 (77.1%) had experiences of using mobile network accessed applications (e.g. SMS) and 796 (71.1%) had that of non-network accessed applications (e.g. Profile). Most of the participants (87.9%) used mobile phone more than half an hour per day. More than half of them (62.1%) used phone more than one hour per day. This indicates that mobile phone usage is quite common and popular in Chinese universities. In addition, SMS usage is very regular and frequent for Chinese university students. 71.4% participants sent or received SMS more than 10 times per day. SMS has become an important part of their university life. This also proves that using SMS as an example mobile application in our experiment is appropriate and easy to be followed by the participants.

3.3 Data Processing and Analysis

We use large sample size to provide better confidence in the results. The samples were randomly divided into two approximately equal parts. One part ($n=567$) was used for the Principle Component Analysis (PCA), while the remaining samples ($n=553$) were used for Confirmatory Factor Analysis (CFA).

Phase 1: Principal Components Analysis (PCA)

Because some items were added and revised according to the results of pre-experiment [20], in the first phase, exploratory, principal components factor analysis and internal consistency reliability analysis were conducted to determine the extent to which trust constructs were discriminant (using SPSS v11.5). The purpose of using PCA was to cull out the items that did not load on the appropriate high-level construct. Kaiser's criterion was applied in the PCA, which considers factors with an eigenvalue greater than one as common factors [30].

Phase 2: Confirmatory Factor Analysis (CFA)

The second phase was a CFA, using Structural Equation Modeling to assess the convergent validity (CV) and discriminant validity (DV) of the latent sub-constructs in each of the three high-level trust constructs (i.e. Using Behavior, Reflection Behavior, and Correlation Behavior). We conducted this analysis by creating a LISREL 8.53 path diagram for each construct, its constituent sub-constructs, and their items. We applied the following indices and criteria to assess model fitness: goodness-of-fit index (GFI) and normed fit index (NFI) greater than 0.90, adjusted goodness-of-fit index (AGFI) greater than 0.80 [22], comparative fit index (CFI) greater than 0.90 [23], and root mean square of approximation (RMSEA) lower than 0.08 for a good fit and lower than 0.05 for an excellent fit [24]. The χ^2 statistic is particularly sensitive to sample size (that is, the probability of model rejection increases with an increasing size of samples, even if the model is minimally false), and hence adjusted $\chi^2(\chi^2/df; df=\text{degrees of freedom})$ is suggested as a better fit metric [25]. It is recommended that this metric should not exceed 5 for a model with good fitness [26].

If the model's fitness is good, we further assess the convergent validity and discriminant validity of the latent sub-constructs inside each of the three high-level trust constructs. Convergent validity was assessed using three criteria: a) individual item lambda coefficients are greater than 0.5; b) t statistic has a significant 0.05 level for each path [22]; and c) each path's loading is greater than the twice of its standard error [27]. Discriminant validity among the latent variables is unquestionless if the inter-correlation between different latent variables is less than 0.6 [28].

3.4 Trust Behavior Construct

A trust behavior construct for mobile applications is achieved based on the above data analysis according to the listed criteria with sound reliability (Using behavior: $\alpha=0.71$; Reflection behavior: $\alpha=0.85$; Correlation behavior: $\alpha=0.79$; overall trust behavior: $\alpha=0.90$), as shown in Fig. 1. Reliability is a value between 0 and 1 expressed by alpha with a larger value indicating better reliability. Generally, alpha above 0.7 implies good reliability [21]. In summary, the using behavior, the reflection

behavior, and the correlation behavior represent the user’s trust behaviors. The PCA, CFA and reliability analysis showed that the questionnaire has positive psychometric properties with respect to construct validity and reliability.

In addition, the relationships of different components in Fig. 1 are set based on the correlations among the three principle factors of trust behaviors and their external variables. We found that all of the three factors had significant correlation with the trust behavior at the 0.01 level, which indicates that these three factors can represent the trust behavior. We also found that these factors had lower correlations with each other than their correlations with the trust behavior. This indicates that these three factors can measure not only the general aspects but also the specific aspects of the trust behavior. Notably, their mutual correlations are around 0.5, which implies that these factors may influence or impact with each other. But the assumed relationships can not be well proved by internal nomological validity of our experiment and in literature theory. This means that these factors could be correspondingly in parallel, without any causal relationships.

We also found the influence of a number of external variables (i.e. personal motivation, brand impact, perceived device quality and personality) on the using behavior, the reflection behavior and the correlation behavior; their correlations are shown in Fig. 1. (Note that ** indicates correlation is significant at the 0.01 level (2-tailed); * indicates correlation is significant at the 0.05 level (2-tailed).)

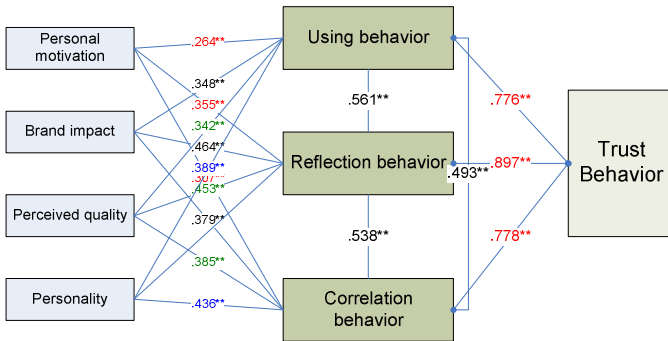


Fig. 1. Trust behavior construct of mobile applications

Furthermore, we illustrate the sub-construct of the using behavior, the reflection behavior and the correlation behavior according to a correlation analysis in Fig. 2, Fig. 3, and Fig. 4, respectively. As can be seen from the figures, the correlation between each internal sub-factor (e.g. UB1, UB2, UB3) and its corresponding principal factor (e.g. UB) is almost in the same level (except CB3’s correlation with CB is a bit lower than CB1 and CB2). This correlation is also higher than the correlations among the sub-factors. This indicates that the sub-factors belonging to a concrete factor can measure not only the general aspects but also the specific aspects of the represented trust behavior.

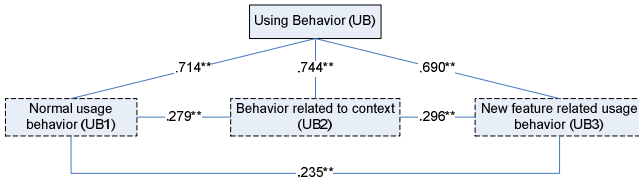


Fig. 2. Internal relationships of Using Behavior

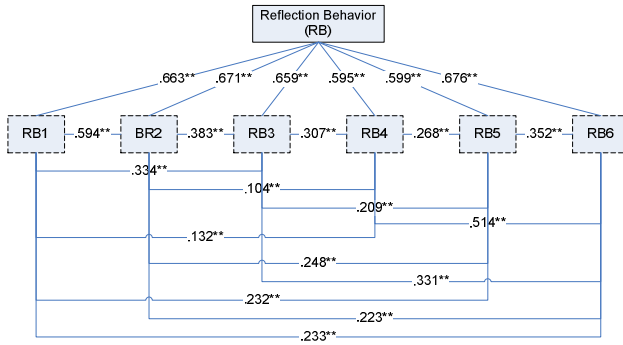


Fig. 3. Internal relationships of Reflection Behavior

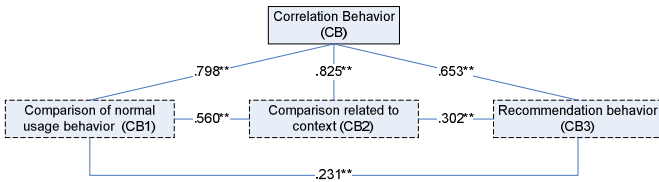


Fig. 4. Internal relationships of Correlation Behavior

4 Trust Formalization

We propose a behavior-based trust evaluation model for mobile applications, which reflects the above trust behavior construct. It includes a coherent adaptive trust model for quantifying and indicating the device’s confidence on user’s trust in a mobile application based on a user-device interaction monitoring system.

This model has two main features. First, it introduces a number of basic parameters and several adaptive factors for computing a user’s trust in a mobile application, namely, the elapsed usage time / the number of usages / the usage frequency of a mobile application, the total number / time / frequency of application usages, recommendation times and recommendation credibility (which is reflected by the current trust value and a context index), and an application context index (e.g. experienced risk, urgency and importance via using the application). Second, it defines a general trust

metric to combine these parameters in order to calculate the trust value that reflects the user's direct trust based on his/her past trust behaviors.

The model provides two tailors to model recommendation credibility. One is using the current trust value of a mobile application recursively as its credibility measure. The other is the average of importance, urgency and risk the user experienced via the mobile application, in short the context index. Meanwhile, it also considers the trust behavior with regard to similar functioned applications. It aims to support a centralized reputation or recommendation system for advertising, distributing and selling the mobile applications. In order to overcome such an attack as a user may make a good recommendation message to a badly performed application or vice versa, we could attach the direct trust value to the reputation or recommendation system, playing as the credibility of the user's contribution to the mobile application's reputation.

4.1 Notations

We apply the following notations in the formalization.

$T_i(t)$	the trust value that the device believes the user has on mobile application i ;
I	the total number of mobile applications in a user's device;
W	the time window applied to collect usage information;
t	the time variable;
$N_i(t)$	the total number of usages of mobile application i within W ;
$N(t)$	the total number of usages of all mobile applications within W ;
$UT_i(t)$	the total elapsed usage time of mobile applications i within W ;
$UT(t)$	the total elapsed usage time of all mobile applications within W ;
$FE(t)$	the usage frequency of all mobile applications in an underlying device within W ;
$FE_i(t)$	the usage frequency of the mobile application i within W ;
$R(t)$	the total number of recommendations for all mobile applications within W ;
$R_i(t)$	the number of recommendations on the mobile applications i within W ;
$F(i)$	the total number of features of the mobile application i ;
$EF_i(t)$	the user experienced number of features of the mobile application i ;
$ci(i, n)$	the context index of the mobile application i regarding the n th usage;
$CI_i(t)$	the context index representing the importance, urgency and risk factor of the mobile application i ; It is the average of importance, urgency and risk value the user experienced via using the mobile application i : $CI(i) = \frac{1}{N_i(t)} \sum_{n=1}^{N_i(t)} ci(i, n)$;
$ac(i, k)$	$\in [0,1]$, the correlation factor indicating the similarity of application i and application k ;
$f(x)$	the Sigmoid function $f(x) = \frac{1}{1 + e^{-x}}$; used to normalize the trust value into $(0, 1)$
$PI_i(t)$	the performance index that indicates an application i 's performance change.

4.2 Formalizing Using Behavior

The PCA assumes that the extracted factors are based on linear combinations. We consider the influence of the number of usages, elapsed usage time, usage frequency

and experienced features on trust according to the trust behavior. We consider those factors' influence on trust according to the total number of usages and elapsed usage time of all applications, the total number of the application features and the usage frequency of all mobile applications in the underlying device.

$$T_i(t)_{UB} = \left(\frac{N_i(t)}{N(t)} * \frac{UT_i(t)}{UT(t)} * \frac{EF(i)}{F(i)} * \frac{FE_i(t)}{FE(t)} \right) * CI_i(t) \quad (1)$$

Meanwhile, we further tailor trust based on the index of importance, urgency and risk, which is the context index (CI) of the mobile application usage. Supposed that importance index (ii), urgency index (ui) and risk index (ri) of the n th mobile application i usage are $ii(i,n)$, $ui(i,n)$ and $ri(i,n)$, then

$$CI_i(t) = \frac{1}{N_i(t)} \sum_{n=1}^{N_i(t)} ci(i,n) = \frac{1}{N_i(t)} \sum_{n=1}^{N_i(t)} (ii(i,n) + ui(i,n) + ri(i,n)) / 3 \quad (2)$$

Note that, through our user study [29], we found that importance rate is highly related to the elapsed usage time, frequency and the number of usages. Since urgency index and risk index are hard to be set in practice, we simplify the formula (1) as below:

$$CI_i(t) = \mu * \left(\frac{N_i(t)}{N(t)} * \frac{UT_i(t)}{UT(t)} * \frac{FE_i(t)}{FE(t)} \right) \quad (2.1)$$

$$T_i(t)_{UB} = \mu * \left(\frac{N_i(t)}{N(t)} * \frac{UT_i(t)}{UT(t)} * \frac{FE_i(t)}{FE(t)} \right) \left(\frac{EF(i)}{F(i)} \right) \quad (3)$$

where μ is a parameter used to adjust the context index.

4.3 Formalizing Reflection Behavior

Our user study on the reflection behavior showed that the change of the elapsed usage time, the number of usages, the usage frequency and the change caused by the CI have influence on trust. We introduce a parameter called performance index (PI) that can be used to reflect application performance according to our user study result about the reflection behavior.

$$PI_i(t) = d_i \{N_i(t) + UT_i(t) + FE_i(t)\} + d_r \{CI_i(t)\} \quad (4)$$

$$\text{Where, } d_i g(t) = \frac{g(t) - g(t - \tau)}{\tau}, \quad (\tau \rightarrow 0); \quad g(t) \text{ is a function of variable } t; \quad (4.1)$$

τ is a time interval applied to measure the changes of behavior and context. For the same reason mentioned in Section 4.2, we simplify the formula (4) as below:

$$PI_i(t) = 2(d_i \{N_i(t) + UT_i(t) + FE_i(t)\}) \quad (5)$$

The contribution of the reflection behavior to trust value can be specified as:

$$T_i(t)_{RB} = PI_i(t) \quad (6)$$

4.4 Formalizing Correlation Behavior

According to the user experiment result, we formalize trust based on the correlation behavior as below. It contains two parts. The first part reflects the comparison of normal usage behavior and the level of context index on similar applications. The second part reflects the recommendation behavior. Herein, we deduct the contribution of the recommendation behavior according to current trust value and context index.

$$T_i(t)_{CB} = \sum_{k=1, k \neq i}^I ac(i, k) * \left\{ \begin{aligned} & \left[\frac{N_i(t) - N_k(t)}{N(t)} + \frac{UT_i(t) - UT_k(t)}{NT(t)} + \frac{FE_i(t) - FE_k(t)}{FE(t)} \right] + \\ & \left[(CI_i(t) - CI_k(t)) + \frac{R_i(t) - R_k(t)}{R(t)} \right] \end{aligned} \right\} + \lambda \frac{R_i(t)}{R(t)} CI_i(t) * T_i(t) \quad (7)$$

Where λ is a parameter that weights the contribution of the recommendation behavior. An importance reason for us to introduce λ is the correlation of CB3 to the CB is lower than CB1's and CB2's correlation to the correlation behavior, as shown in Fig. 4. We use $ac(i, k)$ to indicate the similarity of the application i and k .

4.5 General Trust Metric

The PCA holds an assumptions that the observed data set to be linear combinations of certain basis. Aggregating all above formalization together, we get the following uniformed formula for direct trust evaluation.

$$T_i(t) = T_i(t)_o + \rho T_i(t)_{UB} + \sigma T_i(t)_{RB} + \zeta T_i(t)_{CB} \quad (8)$$

where ρ, σ, ζ denote the normalized weight factors for using behavior evaluation, reflection behavior evaluation, and correlation behavior evaluation.

The metric consists of four parts. The first part is original trust value, which could be an initial trust value at the beginning of the application usage or a trust value generated in the previous time window. The original trust value could be negative since the usage could go down or a user could prefer using another similar application. The second part is a real usage experience based trust evaluation. We consider the percentages of elapsed usage time, frequency and the number of usages, as well as the application features experienced. The third part is contributed by the reflection behavior according to the application's performance which is reflected by usage changes and context index change. The last part is a weighted evaluation contribution about the correlation and recommendation behaviors. The weight takes the current trust value into account to counter dishonest recommendations, and capture the context influence on the recommendations. This history-based evaluation can be seen as a prediction for the recommendation behaviors regarding the trust value contribution. Inside the last part, there is an application-comparison based contribution. This part adjusts the trust value by increase or decrease the first two parts based on the difference of usage number / time / frequency, recommendations and the context index. In order to uniform the trust value in the scope of [0, 1], we apply a sigmoid function on the trust value

$$T_i(t) = f \left\{ T_i(t)_o + \rho T_i(t)_{UB} + \sigma T_i(t)_{RB} + \zeta T_i(t)_{CB} \right\} \quad (9)$$

Important to note is that this general trust metric may have different appearances depending on which of the parameters are switched on and how the parameters and weight factors are set. The setting of $\rho, \sigma, \text{and } \zeta$ could be based on the correlation of UB, RB and CB to trust behavior as 0.776, 0.897 and 0.778.

5 Simulations

From Nokia SmartPhone 360 usage statistics [31], we can figure out one usage model that is periodically changed, e.g. mobile email usage. We use function $|\sin(\omega x)|$, ($\omega=1$) to model it in our simulation with regard to usage frequency. The second usage model could be a logistic function, also known as Richards' curve, which is widely-used for growth modeling. We use a modified logistic function $(1 - e^{-x}) / (1 + e^{-x})$, ($\gamma=1/2$) in our simulation in order to make the growth start from 0 at $t=0$. The third usage model is a growth curve at the beginning and then reducing to a stale level (including 0, which can be controlled by the function parameters). Herein, we use a $\Gamma(\alpha, \beta)$ distribution $t^{\alpha-1} e^{-\beta t}$ ($\alpha=2; \beta=0.5$) to model it. We also propose a linear increase model ηt ($\eta=0.1, \eta < 1$) to roughly model, for example, recommendation percentage, elapsed usage time and the number of usages. The above usage models can be applied in usage time, the number of usage, frequency, or context index. The user experienced feature $EF(i)/F(i)$ could be increased quickly and then gradually stay in a stable level. We use the logistic function to model it.

Fig. 5 (a), (b), and (c) show the simulation results of usage behavior formalization, reflection behavior formalization and correlation behavior formalization, respectively. The usage models (or functions) applied in the simulations are listed in Table 2. For simplification, we apply function (3) and (5) in our simulation. Fig. 5(d) shows the aggregated trust value ($T_i(t_o)=0.5$) based on function (9) and the data of T(UB)_3; T(RB)_3; and T(CB)_1, T(CB)_2, and T(CB)_3, respectively.

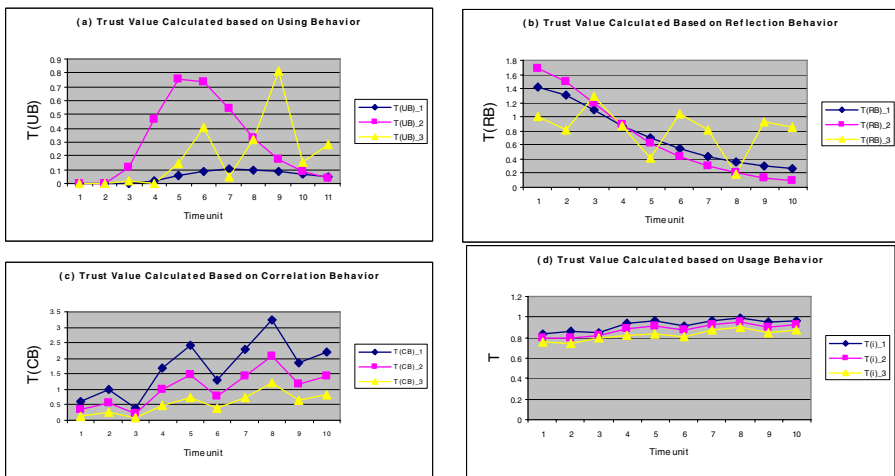


Fig. 5. Trust value calculated based on usage behavior

From the simulation, we can see that the trust value calculated based on the proposed formalization reflects usage change no matter it is periodically up and down or increased or decreased. It also implies the context’s influence on trust. The trust value contributed by the correlation trust behavior indicates the impact of application similarity and usage difference on trust. To uniform the result, we apply a sigmoid function to map final trust value into (0, 1). We can also use this function to map different part of trust contribution into (0, 1) and then aggregate them together. In this case, the general metric becomes:

$$T_i(t) = f\{T_i(t)_o\} + \rho f\{T_i(t)_{UB}\} + \sigma f\{T_i(t)_{RB}\} + \zeta f\{T_i(t)_{CB}\} \quad (\rho + \sigma + \zeta = 1) \tag{10}$$

Table 2. Usage models applied in simulations ($(\eta = 0.1; \gamma = 1/2; \alpha = 2; \beta = 0.5)$)

UB	$\frac{N_i(t)}{N(t)}$	$\frac{UT_i(t)}{UT(t)}$	$\frac{EF(i)}{F(i)}$	$\frac{FE_i(t)}{FE(t)}$	μ	
T(UB)_1; T(RB)_1	η	η	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$t^{\alpha-1} e^{-\beta}$	10	
T(UB)_2; T(RB)_2	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$t^{\alpha-1} e^{-\beta}$	10	
T(UB)_3; T(RB)_3	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$(1 - e^{-\gamma}) / (1 + e^{-\gamma})$	$ \sin(t) $	1	
CB	$\frac{N_i(t) - N_k(t)}{N(t)}$	$\frac{UT_i(t) - UT_k(t)}{NT(t)}$	$\frac{FE_i(t) - FE_k(t)}{FE(t)}$	$CI_i(t) - CI_k(t)$	$\frac{R_i(t) - R_k(t)}{R(t)}$	
Applied functions	$(1 - e^{-\gamma}) / (1 + e^{-\gamma}) - \eta$	$(1 - e^{-\gamma}) / (1 + e^{-\gamma}) - \eta$	$ \sin(t) - t^{\alpha-1} e^{-\beta}$	Apply function (2.1)	η	$\lambda = 1;$ $R_i(t) / R(t) = \eta$
T(CB)_1 T(i)_1	$(ac=0.9; T_i(t)_o=0.5); \rho = \sigma = \zeta = 1$					
T(CB)_2 T(i)_2	$(ac=0.5; T_i(t)_o=0.5); \rho = \sigma = \zeta = 1$					
T(CB)_3 T(i)_3	$(ac=0.2; T_i(t)_o=0.5); \rho = \sigma = \zeta = 1$					

6 Conclusions and Future Work

User-application trust is becoming more and more important for developing and facilitating mobile applications and mobile internet based services. Studying the trust behavior helps greatly in explaining trust status because real behavior based explanation is more convinced. In this paper, we proposed a computational trust model for mobile applications based on users’ trust behaviors. This model is formalized according to a trust behavior construct achieved from a large scale user experiment. This construct has been examined and proved using the principal components analysis, reliability analysis, correlation analysis and confirmatory factor analysis. Based on the results achieved, we got the main factors and construct of trust behavior that contribute to the calculation of the user’s trust in a mobile application. The model formalization was in a computational measure in order to apply the user study result in a practical trust evaluation system (e.g. in a mobile device). We further evaluate our model based on a set of simulated data. The results show that the formalization reflects the trust behavior construct and supports the proved trust behavior measure.

Regarding the future work, we will prototype a secure trust evaluator in a mobile device based on the formalized trust evaluation functions. It should also ensure the user's usage privacy. The formalization functions can be further improved according to the real usage models. We also plan to develop a mobile application reputation system based on the direct trust collected from individual devices. Our research focus will be a robust reputation algorithm that can aggregate the individual direct trust together and defend against potential malicious attacks on our trust model.

Acknowledgement

The authors would like to thank Dr. N. Asokan's valuable comments for the paper improvement.

References

- [1] Marsh, S.: Formalising Trust as a Computational Concept. Doctoral dissertation, University of Stirling (1994)
- [2] Yan, Z., Prehofer, C.: An Adaptive Trust Control Model for a Trustworthy Component Software Platform. In: Xiao, B., Yang, L.T., Ma, J., Müller-Schloer, C., Hua, Y. (eds.) ATC 2007. LNCS, vol. 4610, pp. 226–238. Springer, Heidelberg (2007)
- [3] Daignault, M., Marche, S.: Enabling Trust Online. Proceedings of the Third International Symposium on Electronic Commerce (October 2002)
- [4] Muir, B.M.: Trust in Automation: Part I. Theoretical Issues in the Study of Trust and Human Intervention in Automated Systems. *Ergonomics* 37(11), 1905–1922 (1994)
- [5] Fishbein, M., Ajzen, I.: Beliefs, Attitude, Intention and Behavior: an Introduction to Theory and Research. Addison-Wesley, Reading (1975)
- [6] Lance, J., Hoffman, L.J., Kim, L.J., Blum, J.: Trust Beyond Security: an Expanded Trust Model. *Communications of the ACM* 49(7) (July 2006)
- [7] Yan, Z., Holtmanns, S.: Computer Security, Privacy and Politics: Current Issues, Challenges and Solutions. In: Trust Modeling and Management: from Social Trust to Digital Trust. IGI Global (2008)
- [8] McKnight, D.H., Cummings, L.L., Chervany, N.L.: Initial Trust Formation in New Organizational Relationships. *Acad. Management Rev.* 23(3), 473–490 (1998)
- [9] Bigley, G.A., Pearce, J.L.: Straining for Shared Meaning in Organization Science: Problems of Trust and Distrust. *Acad. Management Rev.* 23(3), 405–421 (1998)
- [10] McKnight, D.H., Choudhury, V., Kacmar, C.: Developing and Validating Trust Measures for E-Commerce: an Integrative Typology. *Information Systems Research* 13(3), 334–359 (2002)
- [11] Li, X., Valacich, J.S., Hess, T.J.: Predicting User Trust in Information Systems: a Comparison of Competing Trust Models. In: Proc. of 37th Annual Hawaii International Conference on System Sciences, January 2004, p. 10 (2004)
- [12] Anderson, J.C., Narus, J.A.: A model of Distributor Firm and Manufacturer Firm Working Partnerships. *Marketing* 54(1), 42–58 (1990)
- [13] Fox, A.: Beyond Contract: Work, Power, and Trust Relations. Faber, London (1974)
- [14] Deutsch, M.: The Resolution of Conflict: Constructive and Destructive Processes. Yale University Press, New Haven (1973)

- [15] Sheppard, B.H., Hartwick, J., Warshaw, P.R.: The Theory of Reasoned Action; A Meta Analysis of Past Research with Recommendations for Modifications in Future Research. *Consumer Res.* 15(3), 325–343 (1988)
- [16] Venkatesh, V., Davis, F.D.: A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies. *Management Sci.* 46(2), 186–204 (2000)
- [17] Muir, B.M.: Trust in Automation Part II: Experimental Studies of Trust and Human Intervention in a Process Control Simulation. *Ergonomics* 39(3), 429–469 (1996)
- [18] Xiong, L., Liu, L.: PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. *IEEE Transactions on Knowledge and Data Engineering* 16(7), 843–857 (2004)
- [19] Lee, J., Moray, N.: Trust, Control Strategies and Allocation of Function in Human-Machine Systems. *Ergonomics* 35(10), 1243–1270 (1992)
- [20] Yan, Z., Niemi, V., Dong, Y., Yu, G.: A User Behavior Based Trust Model for Mobile Applications. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) *ATC 2008*. LNCS, vol. 5060, pp. 455–469. Springer, Heidelberg (2008)
- [21] Crocker, L., Algina, J.: *Introduction to Classical and Modern Test Theory*. Thomson Learning (1986)
- [22] Gefen, D., Straub, D., Boudreau, M.: Structural Equation Modeling and Regression: Guidelines for Research Practice. *Comm. AiS* 7(7), 1–78 (2000)
- [23] Jiang, J.J., Klein, G.: Supervisor Support and Career Anchor Impact on the Career Satisfaction of the Entry-level Information Systems Professional. *Management Inform. Systems* 16(3), 219–240 (1999/2000)
- [24] Browne, M.W., Cudeck, R.: Alternative Ways of Assessing Model Fit. In: BoUen, K.A., Long, J.S. (eds.) *Testing Structural Equation Models*, Sage, Beverly Hills (1992)
- [25] Bentler, P.M., Bonnett, D.G.: Significance Tests and Goodness of Fit in the Analysis of Covariance Structures. *Psychological Bulletin* 88(3), 588–606 (1980)
- [26] Bentler, P.M.: *EQS Structural Equations Program Manual*. BMDP Statistical Software, Los Angeles (1989)
- [27] Anderson, J.C., Gerbing, D.W.: Structural Equation Modeling in Practice: A Review and Recommended Two-Step Approach. *Psychological Bulletin* 103(5), 411–423 (1998)
- [28] Carlson, D., Kacmar, K., Williams, L.: Construction and Initial Validation of a Multidimensional Measure of Work-Family Conflict. *Journal of Vocational Behavior* 56(2), 249–276 (2000)
- [29] Yan, Z.: Trust Indication for Mobile Applications: User Experiment Results (Finland), Nokia Research Center technical report (October 2008)
- [30] Nunnally, J.C.: *Psychometric Theory*, 2nd edn. McGraw-Hill, New York (1978)
- [31] Nokia SmartPhone 360 panel survey results,
<http://nwiki.nokia.com/Smartphone360/WebHome>

Toward Trustworthy Semantic Web Service Discovery and Selection

Jing Li, Dianfu Ma, Jun Han, and Xiang Long

School of Computer Science and Engineering, Beihang University, Beijing, China
{lijing,dfma}@nlscde.buaa.edu.cn,
{jun_han,long}@buaa.edu.cn

Abstract. Accurately locating trustworthy Web Services is one of the most important preconditions of services composition and invocation. However, the rapid growth of Web Services on the Internet makes services discovery become harder. Besides, the open Internet environment prevents us from finding the right services, for the cheating, incorrect and outdated data, which we call “dirty data”, are ubiquitous. To make Web Service consumers get trustworthy services possible, a novel trustworthy Semantic Web Service discovery and selection mechanism is proposed. It uses the consumers’ feedback to describe Web Service’s and service provider’s trustworthy degree which we define as service reputation and service provider reputation. Based on this mechanism and our previous work, an Agent-based Adaptive Dynamic Semantic Web Service Selection framework is presented. According to this framework, a case study is described to show the detailed operating steps of the discovery and selection process.

Keywords: Services, Trustworthy, Semantic Web Service, Service Discovery.

1 Introduction

Web Services were initiated to support the reuse and interoperation of software components on the web, and are receiving ever increasing interests from consumer, industries, and research communities across different domains. A fundamental problem of Web Services concerns service discovery. After the Service discovery step, service consumers face a dilemma in having to making a choice from a bunch of services offering the same function in the Service discovery results. And Web Service selection deals with choosing a service from those that are discovered for the given description. A rapid increase of web services on the Internet [1] makes services discovery and selection become harder. To retrieve services in an accurate and automatic way among a large numbers of candidates, researchers have presented some good efforts on both Service discovery [2-4] and selection [13-16]. Among these contributions, Semantic Web Service has attracted a lot of attention in recent years. Bringing Semantics into Web Service has greatly improved the recall and precision of service discovery.

However, to discover the really useful services for consumers, these discovery efforts have to be based on conditions that the environment which services exist in is

steady and all the services are trustworthy. Otherwise consumers may find that, although the descriptions of services match their requirements perfectly, they cannot invoke these services. Because services are in a changing environment, sometimes they cannot be invoked due to unreachable hosts or failed networks. Further, since we are in a world which has cheats, these services might act as viruses and cause unpredictable loss. Since changing and cheating could not be prevented, mechanisms which can discover the changing and cheating have to be found out.

There are some works [5-11] which address solving this problem. Some of these efforts depend on a QoS registry to collect and store feedbacks from consumers. The general idea of these methods is that consumers report the data acquired from executing a web service and/or their ratings on other QoS metrics to the central QoS registry. Then the consumer will find the web service with the highest rating. Considering the change of QoS is an important way to discover the changing environment, but still it cannot become aware of cheating. Some of these efforts suggest using collaborative filtering technique to avoid cheating, but problems brought by the joining of new users and new items in this technique have not been solved. Some efforts introduce using service reputation to rank services, but no details are provided as to how the reputation score of a service is computed.

We propose a novel trustworthy Semantic Web Service discovery mechanism to adapt this changing and cheating conditions and present an Agent-based Adaptive Dynamic Semantic Web Service Selection (AADSS) framework based on this novel mechanism and our previous work [12]. In our opinion, trustworthy service discovery mechanism includes at least four parts which are trustworthy service advertisements and consumers requirements, trustworthy Services, trustworthy Service providers and trustworthy discovery algorithm. To a semantic web service, although we could not know whether the service provider submits a service advertisement exactly according to the service itself or not, we could judge it by checking if the ontology concepts it refers exist or not to make sure at least it's not a mess-up description file. For the same reason, the requirement has to pass this semantics check too. Service trustiness is hard to scale. We use the consumers' feedback about the invoking successful rate to describe the trustiness, which we call service reputation. We also consider service provider's reputation, which is based on the service reputation evaluated by the consumer. In AADSS framework, we adopt intelligent agents to handle the changing environment and submit feedbacks to the semantic registry. We provide QoS and service reputation subscription method to notify agents these changes.

The remainder of this paper is structured as follows. In Section 2, our trustworthy Semantic Web Service discovery mechanism is introduced in detail. Section 3 outlines the related research conducted in the area of Web services discovery and Section 4 presents a framework based on this mechanism. Finally, Section 5 is the conclusions and our future works.

2 Related Work

There are several efforts focused on trustworthy service and reputation evaluation. The research into establishing trust can be classified into two categories, centralized

and distributed. Most of them depended on a central registry to collect and store feedbacks from consumers. Usually, the feedbacks are according to the QoS that consumer detected. Then, the feedbacks become the foundation of service reputation. The usual ways are based on the hypothesis that, the higher reputation service has, the more trustworthy it is provided with.

Ali et al. [8] designed a policy language, and captured the user's trust disposition in this language, verified the trustworthiness of a service by exploring the reputation of the service and combining it with the user's central view of the service, made a practical decision on whether to trust or distrust a service. Maximilien and Singh [6] proposed a framework which provides a solution to the trust problem by having agents collect and share information on their interactions with the services that they selected on behalf of service consumers. In [5, 6], Maximilien and Singh put a lot of effort on building a QoS ontology, the basis for service providers to advertise their services and for consumers to express their preferences and provide ratings. Liu, Ngu and Zeng [13] proposed an algorithm about how to combine different QoS metrics to get a fair overall rating for a web service. Manikrao and Prabhakar [12] use the collaborative filtering technology in their web service selection method. Majithia et al. [21] proposed a framework for reputation-based semantic service discovery. Ratings of services in different contexts, referring to either particular application domains, or particular types of users, are collected from service consumers by a reputation management system. A coefficient (weight) is attached to each particular context. The weight of each context reflects its importance to a particular set of users. A damping function is used to model the reduction in the reputation score over time. This function, however, only considers the time at which a reputation score is computed, and ignores the time at which a service rating is made. To guarantee that the update of service reputation value can be known by every consumer in time, we adopt reputation subscription policy. Besides, we evaluate the reputation of service provider which could be an effect factor of service reputation.

The only trust and reputation approach for decentralized web service system is proposed by Vu, Hauswirth and Aberer [9]. They use some dedicated QoS registries to collect QoS feedbacks from consumers. Although these QoS registries are organized in a P2P way, they are based on a specially designed P-Grid structure. Each registry is responsible for managing reputation for a part of service providers. An algorithm is introduced to detect and deal with dishonest feedbacks by comparing the QoS data from dedicated monitoring agents with the data from consumers to filter out dishonest feedbacks. This approach is much more complicated than those centralized trust and reputation methods and involves a lot of communication and calculation because of the use of the complicated P-Grid structure.

Our Trustworthy Service Discovery and Selection Mechanism can be adopted by centralized system. We use consumer feed back to evaluate service reputation, and consider service provider's reputation too. Taking both service long-term behave and short-term behave into consideration; we setup two kinds of service reputation. Service long-term reputation is used to calculate its provider's reputation. Service short-term reputation is used to describe its recent behaves.

3 Trustworthy Service Discovery and Selection Mechanism

3.1 Trustworthy Service

Trust as a social phenomenon, as a mental state and as the basis for establishing business alliances and partnership has been widely studied in social network analysis [18], cognitive psychology [19], and economics. More recently, trust is taken as a high level social abstraction of agencies [20] and is studied in the multi-agent research communities. As defined by Wikipedia, “trust is a relationship of reliance. A trusted party is presumed to seek to fulfill policies, ethical codes, law and their previous promises.” But until now, there is no accepted definition of trustiness in Web Service domain. In [17], Wang addressed that, “trust is personalized and subjective reflecting an individual’s opinion”.

In this paper, we give our comprehension of trustworthy service in service discovery mechanism. Trustworthy service is a service which is described in an acknowledged way and contains accepted semantics; it is provided by a high-reputation provider and has believable function and non-function profile. To discover this trustworthy service, we design a trustworthy service discovery mechanism which includes validating the semantic compatibility between ontology and services/requests, evaluating the reputation of services, evaluating the reputation of service providers and a service discovery algorithm based on these factors.

3.2 Validate Semantics Compatibility

In this open environment, everyone can publish services on the public service registry. Some of them maybe have no strict semantics, which means the services are annotated by nonexistent concepts. To avoid these un-trusted services polluting the valid data, we have to validate the semantics compatibility before allowing them to be published. The same thing has to be done before we accept a consumer’s request. In a trustworthy service discovery mechanism, we should avoid un-trusted request taking up available computing resource.

There are several kinds of Semantic Web Service description language standards which have been widely used, for example, SAWSDL, WSDL-S, OWL-S, and so on. Though they have different structures and expressing semantics methods, at least there is one thing in common, that is they use ontology concept to annotate service semantics. Figure 1 shows the segments of SAWSDL, OWL-S, and WSDL-S. Based on this fact, our idea is using ontology to validate services semantics compatibility.

When a service advertisement is submitted, we extract the ontology concepts that it contains, and try to access these ontologies. If the ontology cannot be access or the ontology has incorrect format, this advertisement is invalid. If the concept it refers does not exist in the ontology, this advertisement is also invalid. When a consumer request is submitted, we do the same thing to find out if it is valid or not.


```

...
<wsdl:interface name="CheckAvailabilityRequestService"
  sawsdl:modelReference="http://example.org/Categorization#Electronics">
  ...
</wsdl:interface>
...

```

A segment of SAWSDL

```

...
<wssem:precondition name="ExistingAcctPrecond"
  wssem:modelReference="POOntology#AccountExists">
<wssem:effect name="ItemReservedEffect"
  wssem:modelReference="POOntology#ItemReserved"/>
...

```

A segment of WSDL-S

```

...
<profile:hasInput rdf:resource="&ba_process;#AcctName"/>
<profile:hasInput rdf:resource="&ba_process;#Password"/>
<profile:hasInput rdf:resource="&ba_process;#Confirm"/>
...

```

A segment of OWL-S

Fig. 1. The segments of SAWSDL, OWL-S, and WSDL-S

3.3 Evaluate the Reputation of Services and Service Providers

In general, reputation is the public's opinion about the character or standing (such as honesty, capability, reliability) of an entity, which could be a person, an agent, a product or a service. It is objective and represents a collective evaluation of a group of people/agents. There are still some problems in nowadays reputation evaluation mechanisms. In many reputation systems, the reputation is quantized as a real number, and people tend to choose the service which has highest reputation value. In this case, the new services will not able to get chance and grow up, no matter how good service they offer. To present a fair competition environment, we have to guarantee that all services have their chances to grow up. Service provider's reputation has been ignored by some reputation systems. In fact, service provider's reputation is also important, especially when new services are published. It helps the system to avoid a cold start. Besides, most of the reputation systems only consider QoS in the reputation evaluating process. QoS is an important indicator for service, but we discuss it in the non-function matching step. In our mechanism, we propose a reasonable, operable, and continuable reputation evaluating mechanism, which considers the service providers' reputation, and gives an easy evaluate way to calculate the reputation value. We also offer method to guarantee all the services have their chances to grow up.

The way changes a service's reputation is when one consumer invokes the service successfully, add 1 to its reputation, or else decrease 1. We assign a reputation value to the service according to its provider's services reputation values when it is just published. To service provider, the reputation is decided by all the services it provides.

If the reputation value of provider A is R_p , and A has published m services, the reputation of every service is R_{w1} , R_{w2} ... R_{wm} . At this time, provider publishes a new service w_{m+1} . Formula 1 and 2 show how to calculate the reputation of w_{m+1} . In

formula 1, fre_i is a percentage that shows the invoked frequency of service w_i ($1 \leq i \leq m$), $times_i$ is the number of times that w_i has been invoked. In formula 2, $R_{w_{m+1}}$ is the reputation of service w_{m+1} , R_{w_i} is the reputation of service w_i . Formula 3 shows how to calculate the provider A's reputation R_p . We consider the time element when we evaluate a provider's reputation. The variable $Time$ is the existent time of this provider. In one particular system, $Time$ can be any unit, for example, second, minute, hour, and so on. R_p stands the average reputation of all the services owned by one provider during the time it exists. It's a comparable value between providers.

$$fre_i = \frac{times_i}{\sum_{j=1}^m times_j} \quad (1)$$

$$R_{w_{m+1}} = \sum_{i=1}^m fre_i \times R_{w_i} \quad (2)$$

$$R_p = \sum_{i=1}^m fre_i \times R_{w_i} / Time \quad (3)$$

When a new service provider is published, to avoid the cold start, we assign a reputation value to this provider. This value equals the average reputation value of all the service providers in the service registry. And when a provider publishes its first service, we assign the average reputation of all the services as its initial value.

Because services' reputation can accumulate as time goes by, the services which exist longer always have more chance to get higher reputation. To guarantee all the services have chances to grow up, we setup two kinds of service reputation for one service. One is R_w which we have discussed at the fore, we call it long-term reputation. It's used to evaluate the provider's reputation. The other one R_{ws} is called short-term reputation. Then we set a time-limit, for example, one month. R_{ws} is the reputation value that is accumulated in the last time-limit. According to this evaluate method, the reputation value of services and providers can be any rational number.

The provider is organized as different levels according to their reputation value. When consumers look for some service, they can consider about choosing the provider level first and then pick up one service according to R_{ws} .

3.4 Trustworthy Service Discovery and Selection Algorithm

In this paper, we name a "Trustworthy Service Discovery and Selection Algorithm" as an algorithm that matches trustworthy service advertisements and consumer requirements according to their exact semantics and select the "right" service based on the trustworthy service reputation.

First, we validate the semantics compatibility of the inputs of this algorithm (service advertisements and consumer requirement), then we perform an exact service discovery algorithm to find the candidate service list. At last, we select one service from this list according to services' reputation value. The service discovery algorithm is in figure 2 and 3. The algorithm consists service function matching and service non-function matching, and currently only QoS is considered in the non-function matching part.

Algorithm 1. Service Discovery by Function

```

function ServiceDiscovery (BLq, q)
for each wi in BLq
  if (wi.OperationConcept  $\supseteq$  q. OperationConcept)
    if (wi.EffectConcept  $\supseteq$  q. EffectConcept) and (wi.PreconditionConcept  $\supseteq$  q. PreconditionConcept)
      for each q.OutputConceptj in q
        exist wi.OutputConceptj in wi
          If!(wi.OutputConceptj  $\supseteq$  q.OutputConceptj)
            return FALSE;
      for each wi.InputConceptj in wi
        exist q.InputConceptj in q
          If!(wi.InputConceptj  $\supseteq$  q.InputConceptj)
            return FALSE;
    CLq .add(wi);
    return TRUE;
else return FALSE;

```

Parameters:BL_q: The service advertisement set in service registry.w_i: One service advertisement in BL_q.

q: One requirement.

CL_q: The Service set that satisfy consumer's function requirement**Fig. 2.** Service Discovery by Function**Algorithm 2.** Service Discovery by Non-function

```

function QoSFiltering (CLq, q)
for each wi in CLq
  for each q.QoSConcept in q
    exist wi. QoSConcept = q. QoSConcept
    if (wi. QoSConcept.Unit(Maxvalue) < q. QoSConcept.Unit(Maxvalue))
      if (wi. QoSConcept.Unit(Minvalue) > q. QoSConcept.Unit(Minvalue))
        Result.add(wi);
        return TRUE;
    return FALSE;

```

Parameters:CL_q: The service advertisement set in which all the services make Algorithm 1 return True.w_i: One service advertisement in CL_q.

q: One requirement.

Result: The Service set that satisfy consumer's function and non-function request

Fig. 3. Service Discovery by Non-function

In the function matching algorithm, the symbol “ \supseteq ” express the containing relationship between ontology concepts. $A \supseteq B$ means concept B is concept A's equivalent class or super class. “OperationConcept” stands for the concept that is used to annotate the operation. As the same way, “EffectConcept”, “preconditionConcept”, “OutputConcept” and “InputConcept” stand for concepts used to annotate effect, precondition, output and input separately. In the non-function matching algorithm, “QoSConcept” stands for concept used to annotate QoS. “Unit” is the unit of each QoS metric. “Maxvalue” and “Minvale” stand for the maximal value and minimal value that the QoS metric can be.

4 AADSS Framework

4.1 AADSS Framework Overview

We present a novel Agent-based Adaptive Dynamic Semantic Web Service Selection framework AADSS based on this trustworthy service discovery and selection Mechanism. Figure 4 is the structure of AADSS framework which shows the interactions between the main entities.

There are two kinds of agents, one is for service provider called provider-agent, and the other is for service consumer called consumer-agent. Provider-agent extracts the semantic information from semantic Web Service descriptions which is submitted by service providers according to the definition of service publication template, and validates its semantics compatibility before invoking the published API of AADSS registry to publish services. Any semantic Web Service description language will be accepted by our framework, if it contains the information in figure 5. Consumer-agent gets the semantic service request from consumer, and validates its semantics compatibility before invoking service finding API to get a list of services, and subscribes to non-functional service information and service reputation value from AADSS registry. Consumer-agent also filters unfit services based on QoS properties and picks the service with maximal reputation value for consumer. AADSS registry is the core component of this framework. It maintains and manages service semantic descriptions, service providers' information and service reputation values, provides service publishing and finding functions, executes semantic service discovery algorithm, non-functional service description and reputation updating and subscribing algorithm. Main contributions of AADSS are summarized as follows:

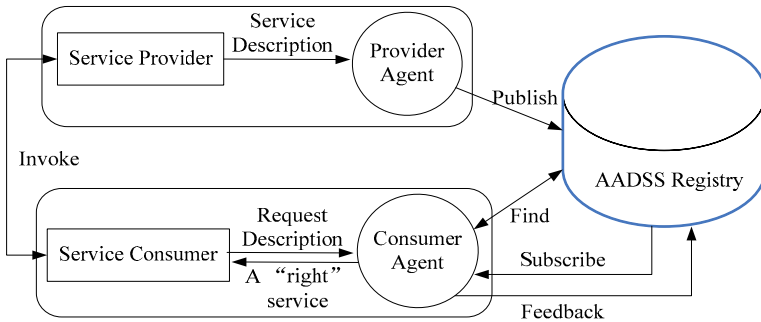


Fig. 4. AADSS Framework

1. Semantics Web Service discovery. We adopt trustworthy service discovery algorithm (present in figure 2) to generate the candidate services list.
2. QoS measure and feedback. We introduce a consumer feedback mechanism to update the quality of services. Consumer-agent filtrates to the candidate service list according to service non-function discovery algorithm (present in figure 3).

```

<service modelReference="placeholder">
  <provider>provider information</provider>
  <wsdl_url>url for wsdl file</wsdl_url>
  <semantic_description>url for semantic web service description file</semantic_description>
  <operation name="placeholder" modelReference="placeholder">
    <input name="placeholder">
      <modelReference>modelReference</modelReference>*
    </input>*
    <output name="placeholder">
      <modelReference>modelReference</modelReference>*
    </output>*
    <precondition> modelReference of predondition</precondition>*
    <effect>modelReference of effect</effect>*
    <qos modelReference="placeholder">
      <unit> modelReference of Unit </unit>
      <maxvalue>Maxvalue</maxvalue>
      <minvalue>Minvalue</minvalue>
    </qos>*
  </operation>*
</service>

```

Fig. 5. Semantic Publication Template

3. Service providers and services reputation evaluation. Adopting user feedback mechanism to calculate the reputation of service providers and services, and updating the reputation value according to Section 3.3. Consumer's agent selects the service which has the maximum reputation value in the candidate list as the "right" service.
4. QoS and service reputation subscription. When the QoS and reputation values are changed, the consumer-agent redoes the QoS filtering work and reputation ranking work. Once the consumer invokes the service, the "right" service is on the top of the list.

4.2 Case Study

In order to test-drive our framework, we apply it to a simple case study from the client's perspective. It supposes that programmer Jing wants to invoke an addition operation in her application, and she chooses to use a Web Service to do the addition work. The process from consumer requesting to service invoking follows these steps:

Step 1

Jing writes her request according to our semantic request template (SRT), and submits to the consumer-agent. Figure 6 is a filled SRT, and the italics are information filled in by Jing. In implementation, we can give a GUI which allows users to fill the parameters they want, and generate this xml automatically. In this simple example, Jing gives two inputs, one output and one QoS metric.

Step 2

The consumer-agent extracts the semantic description from this request, validates its semantics compatibility and then calls the finding API of registry.

Step 3

AADSS registry matches the request to the services in the registry according to the semantically annotated function parameters, and returns the list of matched service that is named candidate service list CL, which includes service invoking information, QoS description and service reputation values.

```

<operation name="Addition" >
  <input name="addend">
    <modelReference>http://sweet.jpl.nasa.gov/ontology/numerics.owl#NumericalEntity</modelReference >
  </input>
  <input name="summand">
    <modelReference>http://sweet.jpl.nasa.gov/ontology/numerics.owl#NumericalEntity</modelReference >
  </input>
  <output name="sum">
    <modelReference>http://sweet.jpl.nasa.gov/ontology/numerics.owl#NumericalEntity</modelReference >
  </output>
  <qos modelReference="http://www.comp.lancs.ac.uk/owl/qos/qosont2.owl#MaximumTimeToComplete">
    <unit>http://sweet.jpl.nasa.gov/ontology/time.owl#Second</unit>
    <maxvalue>0.3</maxvalue>
    <minvalue> 0</minvalue>
  </qos>
</operation>

```

Fig. 6. An Example of Semantic Request

Step 4

Consumer-agent filters services in CL with the QoS requirement “MaximumTimeToComplete”. If the service maximum complete time is smaller than 0.3 second, then agent puts this service into list CL’.

Step 5

Consumer-agent sorts all the services in CL’ by service reputation level and selects the service A which has the maximal short-term reputation value R_{ws} among the highest level providers’ services. Then A is bound and invoked by Jing’s application.

Step 6

After the successful invocation of A, consumer-agent submits the real service quality property value to the registry, which means sending the complete time to registry here. Registry updates the “MaximumtimeToComplete” value of A according to its strategy, and adds 1 to A’s reputation value. If the invocation is unsuccessful, consumer-agent will not submit the QoS property value. Registry subtracts 1 from A’s reputation value.

Step 7

Consumer-agent maintains services information in CL, and subscribes their QoS and reputation value to registry. When the service quality or reputation in CL is changed, registry will notify the consumer-agent about the update of the correlative information. Before every invocation, consumer-agent will redo step 4 to step 6, unless one of these four events happens: Jing changes the request; she claims to update the candidate service list; the invocation failed; all the services in CL cannot be filled to the QoS requirements. In these four situations, consumer-agent will redo step 3 to step 6.

5 Conclusions and Future Works

We have described a trustworthy service discovery and selection mechanism which gets the really useful service that consumer wants from the ubiquitous cheating, incorrect and outdated data. We have also designed an agent-based adaptive dynamic Semantic Web Service selection framework based on this mechanism.

Future work will be enhancing the performance of the semantic Web Service discovery algorithm, so that the algorithm can deal with the massive volumes of data online. Since centralized registry has the single point of failure problem and has processing capacity limitation, the AADSS framework will be redesigned as a distributed structure.

Acknowledgments. The authors acknowledge the support from National High Technology Research and Development Program of China (Project No. 2006AA01Z19A) and National Natural Science Foundation of China (NSFC grant number: 60842008 and 90818028).

References

1. Al-Masri, E., Mahmoud, Q.H.: Investigating Web Service on WWW. In: 17th International World Wide Web Conference, pp. 795–804. ACM Press, Beijing (2008)
2. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 36–47. Springer, Heidelberg (2002)
3. Schmidt, C., Parashar, M.: A Peer-to-Peer Approach to Web Service Discovery. *World Wide Web Journal* 7(2), 211–229 (2004)
4. Al-Masri, E., Mahmoud, Q.H.: Discovering the Best Web Service. In: 16th International World Wide Web Conference, pp. 1257–1258. ACM Press, New York (2007)
5. Maximilien, E.M., Singh, M.P.: Conceptual Model of Web Service Reputation. *ACM SIGMOD Record* 31(4), 36–41 (2002)
6. Maximilien, E.M., Singh, M.P.: Toward Autonomic Web Services Trust and Selection. In: 2nd international conference on Service oriented computing, pp. 212–221. ACM Press, New York (2004)
7. Liu, W.: Trustworthy Service Selection and Composition - Reducing the Entropy of Service-oriented Web. In: 3rd IEEE International Conference on Industrial Informatics, pp. 104–109. IEEE Press, Perth (2005)
8. Ali, S.A., Ludwig, S.A., Rana, O.F.: A Cognitive Trust-Based Approach for Web Service Discovery and Selection. In: 3rd European Conference on Web Services, pp. 38–49. IEEE Press, Vaxjo (2005)
9. Vu, L., Hauswirth, M., Aberer, K.: QoS-based service selection and ranking with trust and reputation management. In: Meersman, R., Tari, Z. (eds.) CoopIS/DOA/ODBASE 2005. LNCS, vol. 3761, pp. 466–483. Springer, Heidelberg (2005)
10. Manikrao, U.S., Prabhakar, T.V.: Dynamic Selection of Web Services with Recommendation System. In: 1st International Conference on Next Generation Web Services Practices, pp. 117–123. IEEE Press, Seoul (2005)

11. Xu, Z., Martin, P., Powley, W., Zulkernine, F.: Reputation-Enhanced QoS-based Web Services Discovery. In: 5th International Conference on Web Services, pp. 249–256. IEEE Press, Salt Lake City (2007)
12. Li, J., Ma, D.F., Li, L., Zhu, H.: AADSS: Agent-based Adaptive Dynamic Semantic Web Service Selection. In: 4th International Conference on Next Generation Web Services Practices, pp. 83–89. IEEE Press, Seoul (2008)
13. Liu, Y., Ngu, A., Zheng, L.: QoS computation and policing in dynamic web service selection. In: 13th International World Wide Web Conference, pp. 66–73. ACM Press, New York (2004)
14. Sreenath, R.M., Singh, M.P.: Agent-Based Service Selection. In: Web Semantics: Science, Services and Agents on the World Wide Web, vol. 1(3), pp. 261–279. Elsevier Press, Amsterdam (2004)
15. Hwang, S.Y., Lim, E.P., Lee, C.H., Chen, C.H.: On Composing a Reliable Composite Web Service: A Study of Dynamic Web Service Selection. In: 5th International Conference on Web Services, pp. 184–191. IEEE Press, Salt Lake City (2007)
16. Lamparter, S., Ankolekar, A., Studer, R., Grimm, S.: Preference-based Selection of Highly Configurable Web Services. In: 16th International World Wide Web Conference, pp. 1013–1022. IEEE Press, Salt Lake City (2007)
17. Wang, Y., Vassileva, J.: Toward Trust and Reputation Based Web Service Selection: A Survey. *Multi-agent and Grid Systems Journal* (2007)
18. Burt, R.S.: Bandwidth and Echo in Networks and Markets. In: *Contributions from Economics and Sociology*. Russell Sage Foundation, Thousand Oaks (2001)
19. Castelfranchi, C., Falcone, R.: Principles of trust for AMS: cognitive anatomy, social importance, and quantification. In: 3rd International Conference on Multiagent Systems (1998)
20. Castelfranchi, C.: Commitments: From individual intentions to groups and organizations. In: *The International Conference on Multiagent Systems* (1995)
21. Majithia, S., Ali, A.S., Rana, O.F., Walker, D.W.: Reputation-based semantic service discovery. In: 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 297–302. IEEE Press, Modena (2004)

Fuzzy Regression Based Trust Prediction in Service-Oriented Applications

Lei Li, Yan Wang, and Vijay Varadharajan

Department of Computing
Macquarie University
Sydney, NSW 2109
Australia

{leili, yanwang, vijay}@science.mq.edu.au

Abstract. Reputation-based trust evaluation is critical to e-commerce or e-service applications. In some applications (such as eBay), the trust management mechanisms have been introduced to provide valuable information to buyers prior to placing orders and making payments. Meanwhile, the trust issue is also actively studied in the research community. However, in most existing studies, a single trust value is computed based on ratings given for a seller or a service provider to indicate the current trust level. This is interesting but too simple to reflect the service quality history and trust features well under some circumstances. It may also be misleading for the decision making of buyers or service customers. In this paper, we present a novel fuzzy regression based trust vector approach to depict the trust level with more indications, and predict the trustworthiness of a forthcoming transaction with its advertised QoS values and transaction history.

1 Introduction

In e-commerce or e-service environments, the reputation-based trust is a very important indication for a buyer or a service customer to make the selection from a few sellers or service providers providing the same product or service, since the buyer or service customer would like to order from the seller or service provider with the best transaction reputation. This is particularly important when the buyer or service customer has to select from unknown sellers or service providers.

In a trust management mechanism enabled system, buyers or service customers can provide feedback and ratings after transactions. Then the trust management system can evaluate the trust value based on collected ratings reflecting the quality of recent transactions. The trust value can be provided to buyers or service customers by publishing it on web or responding to their requests.

The trust issue is also actively studied (e.g. [13]) in Peer-to-Peer (P2P) networks, which can be used for information-sharing systems¹. In a P2P system,

¹ <http://www.gnutella.com>

it is quite natural for a client peer to doubt if a serving peer can provide the complete file prior to any download action, which may be quite time-consuming and network bandwidth-consuming. Different from some trust management systems in e-commerce environments, in the P2P trust system, a requesting peer needs to request the trust data of a serving peer (target peer) from other peers which may have transacted with the serving peer [4,8,14]. The computation of the trust status of the serving peer from the collected trust ratings is then performed by the requesting peer, not a central management server, because of the decentralized architecture of P2P systems.

In most existing trust evaluation studies, a single final trust value (e.g. in the range of $[0, 1]$ [4,8,11,13,14,16]) is computed to reflect the *general* or *global* trust level of a seller or service provider. For example, if the final trust value is in the range of $[0, 1]$, 0.95 indicates a very good trust level for a seller or a service provider.

Such a value can reflect the transaction/service reputation accumulated in a certain period. However, a single final trust value may not reflect the real trust status very well. For example, if $T_A = T_B = 0.7$ (in the scale of $[0,1]$), does it indicate that sellers or service providers A and B have the same trust level? It is not true if A 's trust values are becoming worse with an accumulated value of 0.7 while B 's values are becoming better. Thus B is better than A in predicting the trust level in a forthcoming transaction. In general, the single-value approach cannot reflect (i) the service trust trend of changes and (ii) the service performance consistency level. *Service trust trend* indicates that the service trust will become better or worse in forthcoming transactions. This is an important indication to tell buyers or service customers to what extent a seller or a service provider is trustworthy for new transactions or services. *Service performance consistency level* can indicate whether the service quality is being maintained at the level reflected by the final trust value, which makes sense no matter whether the trust value is low or high. For example, if the trust level of a service provider is *good* and its trust level is "*consistent*", that indicates that the service provider has maintained the *good* level for a certain period.

In a good trust management system, it requires more comprehensive trust evaluation approaches providing more (precise) trust information that indicates not only the global trust level, but also the trust prediction *relevant to forthcoming transactions*. To serve this purpose, according to fuzzy regression, in this paper, we propose a *service trust vector* consisting of a set of values, such as *final trust level*, *service trust trend* and *service performance consistency level*, which is applicable to e-commerce or e-service environments. We will also conduct empirical studies to study the properties of our proposed approaches.

In this paper, we adopt fuzzy regression instead of classical regression because in classical regression analysis, the deviations between the observed and estimated data are assumed to be subject to random errors. However, these deviations are frequently caused by the indefinite structure of a system or imprecise observations. All make it necessary to introduce the fuzzy regression.

This paper is organized as follows. In Section 2, we review the trust management approaches of eBay, some existing studies and the fuzzy regression method. In Section 3, some parameters are introduced to prepare for Section 4, which explains the fuzzy regression methodology. Section 5 discusses the *final trust level*, *service trust trend* and *service performance consistency level* evaluations in our service trust vector. Some empirical studies are presented in Section 6 for further illustrating the properties of our model. Finally Section 7 concludes our work.

2 Background

2.1 Trust Management at eBay

The trust management mechanism in eBay² is one of the earliest systems in applications.

At eBay, after each transaction, the buyer can give feedback to the system according to the service quality of the seller. The feedback (or rating) is stored by eBay (a centralized management architecture), which can be “positive”, “neutral” or “negative”. eBay calculates the feedback score $S = P - N$, where P is the number of positive feedback left by members (customers) and N is the number of negative feedback from members. Then S value can be displayed on the web pages. In addition, $R = \frac{P-N}{P+N}$ (e.g., $R = 99.1\%$) is called positive feedback rate, based on which a seller can be awarded as a “Power Seller” if $R \geq 98\%$ (98% is the threshold).

eBay also improves its trust service and provides rating data in 12 months listed in a table, which is divided by recent *1 month*, *6 months* and *12 months*. Thus, eBay provides some simple mechanisms of trust management and trust calculation and leaves some raw data to buyers for self-calculation.

2.2 Trust Management in Other Environments

In [16], a trust evaluation approach is introduced for e-commerce applications which is based on the trust values of transactions in a recent period, rather than all of them. In this method, recent ratings are more important in the trust evaluation. In [9], fuzzy logic is applied to trust evaluation, which divides sellers or service providers into multiple classes of reputation ranks (e.g. a 5-star seller, or a 4-star seller).

Trust is also an important issue in Peer-to-Peer (P2P) information-sharing network⁴, since a client peer needs to know prior to download actions which serving peer can provide complete files. P2P trust evaluation relies on a polling algorithm (e.g. [1]), a binary rating system for calculating the global trust value of a given peer [4,14], or a voting reputation system (e.g. [8]) that calculates the final trust value combining the values returned by responding peers and the requesting peer’s experience with the given peer.

² <http://www.ebay.com>

We know [15] binary rating systems work very well for file-sharing systems where a file is either the definitive correct version or wrong. But binary ratings cannot accurately model richer services such as web services or e-commerce, since a binary rating may not adequately represent a client peer's experience of the quality of service (QoS) with other serving peers, such as the quality of products the serving peer sends and the expected delivery time [15]. In most later studies on trust evaluation (e.g. [12,13,15]), a numeral rating system is adopted, where, for example, the rating is a value in the range of $[0, 1]$. Such a rating system is more suitable for complex applications.

In the literature, trust issue also receives much attention in service-oriented computing research. In [7], Lin *et al* propose a method of reputation-based trust evaluation in service-oriented environments based on a proposed architecture consisting of distributed trust management brokers. In [10], Vu *et al.* present a model to evaluate and rank the trust and reputation of QoS-based services. In [11], an event-driven and rule-based trust management for service-oriented application is proposed, where a formula based approach is adopted for incremental trust computation. Moreover, the approach is adaptable to applications by incorporating rule management. Then, the computed result can be taken as a global trust value reflecting the accumulated trust level, which is not particularly relevant to new transactions.

2.3 Fuzzy Regression

In e-service environments, the service provider usually provides the QoS values before a transaction, which are the advertised QoS values, and then receives the aggregated quality value in $[0, 1]$ about the delivered QoS values after the transaction. Such a delivered quality value is assumed to be able to indicate the success possibility or trustworthiness of this transaction, therefore this delivered quality value can be taken as a trust value. Assuming both the service providers and service consumers are honest, if the transaction history data and the new advertised QoS values are already known, the new delivered quality value can be predicted, since the delivered quality value is inherently related with the corresponding existing advertised QoS values prior to the new transaction.

For example, at eBay, we consider to buy a new battery for HP pavilion dv2000 and dv6000 laptops³. First of all, we focus on the transaction history and try to find out whether the seller is trustworthy. Before each transaction, the seller provides the product's price, shipping price, delivering time and all other QoS values. After the transaction, the buyer gives a feedback rating about the transaction quality. If a function between the advertised QoS values and the delivered rating can be determined, then with the new QoS values, such as: Price US \$58.75, Shipping US \$10.95 and so on, the new rating can be predicted. That is very useful and important before the transaction.

³ http://cgi.ebay.com/NEW-Battery-for-Hp-pavilion-dv2000-dv6000-V6000-12-cell0QQitemZ370143793214QQcmdZViewItemQQptZLH_DefaultDomain_0?hash=_Witem370143793214&_trksid=p3286.c0.m14&_trkparms=72%3A1234%7C66%3A2%7C65%3A12%7C39%3A1%7C240%3A1308%7C301%3A1%7C293%3A1%7C294%3A50

Regression analysis [2] is a statistical technique for modeling and investigating the relationship between two or more variables. For example, here regression analysis can be used to build up a model that represent the delivered quality value as a function of a set of advertised QoS values. Then this model can be used to predict the new delivered quality value with a new set of advertised QoS values.

In the classical regression method, a set of parameters of an unknown function $f(x, \omega)$ can be estimated by making measurements of the function with error at any point x_i :

$$y_i = f(x_i, \omega) + \epsilon_i, \tag{1}$$

where the error ϵ_i is independent of x and is distributed according to a known density $p_\omega(\epsilon)$. Based on the observed data sample $S = \{(x_i, y_i), i = 1, 2, \dots, n\}$, the likelihood is given by

$$P(S|\omega) = \prod_{i=1}^n \ln p_\omega(y_i - f(x_i, \omega)). \tag{2}$$

Assuming that the error is normally distributed with mean 0 and variance δ , the likelihood is given by

$$P(S|\omega) = -\frac{1}{2\delta^2} \sum_{i=1}^n (y_i - f(x_i, \omega))^2 - n \ln(\sqrt{2\pi}\delta) \tag{3}$$

Maximizing the likelihood in Eq. (3) is equivalent to minimizing

$$E(\omega) = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i, \omega))^2, \tag{4}$$

which is in fact the same as the estimation by the method of least squares. Namely, the regression line is estimated so that the sum of squares of the deviations between the observations and the regression line is minimized.

In classical regression analysis, the deviations between the observed and estimated data are assumed to be due to random errors. However, frequently these deviations are caused by the indefinite structure of the system or the imprecise observations, which makes *it necessary to introduce the fuzzy regression*.

Fuzzy regression can be quite useful in estimating the relationships among variables where the available data are very limited and imprecise, and variables are interacting in an uncertain, qualitative, and fuzzy way. Thus, it may have considerably practical applications in many management and engineering problems.

3 Fuzzy Regression Model Parameters

Prior to presenting the detailed fuzzy regression model, some definitions about parameters of the model should be introduced firstly in this section.

3.1 Membership Function

The fuzzy number mentioned in this paper is $\tilde{A}(\alpha, C)$ with the following membership function [3],

$$\mu_{\tilde{A}}(x) = \begin{cases} \begin{cases} 1 - \frac{|x-\alpha|}{C}, & |x-\alpha| \leq C, \\ 0, & \text{otherwise,} \end{cases} & C > 0, \\ \begin{cases} 1, & x = \alpha, \\ 0, & \text{otherwise,} \end{cases} & C = 0, \end{cases} \quad (5)$$

According to Eq. (5), it is easy to prove that $\lambda\tilde{A}$ is fuzzy number $\tilde{A}(\lambda\alpha, |\lambda|C)$ and $\tilde{A}_1 + \tilde{A}_2$ is fuzzy number $\tilde{A}(\alpha_1 + \alpha_2, C_1 + C_2)$. So if

$$\tilde{T}_i^* = \tilde{A}_0 + q_{i1}\tilde{A}_1 + \dots + q_{in}\tilde{A}_n, \quad (6)$$

then \tilde{T}_i^* is the fuzzy number

$$\tilde{T}_i^*(\alpha_0 + \sum_{j=1}^n q_{ij}\alpha_j, C_0 + \sum_{j=1}^n |q_{ij}|C_j). \quad (7)$$

3.2 Goodness-of-Fit

Definition 1. Let A, B be the fuzzy sets in real space \mathbf{R} , then

$$h = \bigvee_{x \in \mathbf{R}} \{ \mu_{\tilde{A}}(x) \wedge \mu_{\tilde{B}}(x) \} \quad (8)$$

is the *goodness-of-fit* from A to B .

In fact, the goodness-of-fit is defined as the inner product here. According to Definition 1, the goodness-of-fit from fuzzy number $\tilde{A}(\alpha_1, C_1)$ to fuzzy number $\tilde{B}(\alpha_2, C_2)$ is

$$h = \begin{cases} 1 - \frac{|\alpha_1 - \alpha_2|}{C_1 + C_2}, & |\alpha_1 - \alpha_2| \leq C_1 + C_2, \\ 0, & \text{otherwise,} \end{cases} \quad (9)$$

Hence, the goodness-of-fit h_i from $\tilde{T}_i(T_i, e_i)$ to Eq. (7) $\tilde{T}_i^*(\alpha_0 + \sum_{j=1}^n q_{ij}\alpha_j, C_0 + \sum_{j=1}^n |q_{ij}|C_j)$ is

$$h_i = \begin{cases} 1 - \frac{|T_i - (\alpha_0 + \sum_{j=1}^n q_{ij}\alpha_j)|}{C_0 + \sum_{j=1}^n |q_{ij}|C_j + e_i}, & |T_i - (\alpha_0 + \sum_{j=1}^n q_{ij}\alpha_j)| \leq C_0 + \sum_{j=1}^n |q_{ij}|C_j + e_i, \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

3.3 Fuzziness

Definition 2. Let $A(\alpha, C)$ be the fuzzy number, then the *fuzziness* [5] of A is

$$S_A = \frac{1}{2}C. \quad (11)$$

The fuzziness measures how fuzzy or vague the fuzzy set is or how clear it is not. According to Definition 2, the fuzziness of \tilde{T}_i^* in Eq. (6) is

$$S_{\tilde{T}_i^*} = \frac{1}{2} (C_0 + \sum_{j=1}^n |q_{ij}| C_j). \tag{12}$$

4 Fuzzy Regression Methodology

Let sample data be

$$\begin{aligned} & q_{11}, q_{12}, \dots, q_{1n}; T_1 \\ & q_{21}, q_{22}, \dots, q_{2n}; T_2 \\ & \dots \\ & q_{m1}, q_{m2}, \dots, q_{mn}; T_m \end{aligned} \tag{13}$$

where $q_{i1}, q_{i2}, \dots, q_{in}$ are the advertised QoS values at time i , which are the input data. T_i is the corresponding delivered quality value, which is the output data, and $i = 1, 2, \dots, m$.

Since there may be no established relation between the input and the output, in order to dovetail the model nicely with the real application, the output is transformed into fuzzification, which makes the fuzzy output \tilde{T}_i be fuzzy number $\tilde{T}_i(T_i, e_i)$, where e_i depends on the application environment. In this paper, the relation between the input and the output is estimated by the fuzzy linear regression as follows.

The corresponding general fuzzy linear regression model is

$$\tilde{T}_i^* = \tilde{A}_0 + q_{i1} \tilde{A}_1 + \dots + q_{in} \tilde{A}_n, \tag{14}$$

where \tilde{A}_j is fuzzy number $\tilde{A}_j(\alpha_j, C_j)$ which has the membership function in Eq. (5) and \tilde{T}_i^* is the fuzzy number defined in Eq. (7).

For parameters estimation, when the goodness-of-fit h_i is large enough, we try to minimize the fuzziness

$$\max_{1 \leq i \leq m} \left\{ \frac{1}{2} (C_0 + \sum_{j=1}^n |q_{ij}| C_j) \right\} \tag{15}$$

to estimate $\alpha_0, \alpha_1, \dots, \alpha_n$ and C_0, C_1, \dots, C_n .

Since

$$\max_{1 \leq i \leq m} \left\{ \frac{1}{2} (C_0 + \sum_{j=1}^n |q_{ij}| C_j) \right\} \leq \frac{1}{2} C_0 + \frac{1}{2} \sum_{j=1}^n (\max_{1 \leq i \leq m} |q_{ij}|), \tag{16}$$

parameters estimation is transformed to linear programming

$$\min S = W_0 C_0 + W_1 C_1 + \dots + W_n C_n, \tag{17}$$

such that

$$h_i > H, i = 1, 2, \dots, m, \tag{18}$$

$$C_j \geq 0, j = 0, 1, \dots, n, \tag{19}$$

where H is established at the beginning, and

$$W_0 = \frac{1}{\sum_{k=1}^n (\max_{1 \leq i \leq m} |q_{ik}|)}, \tag{20}$$

$$W_j = \frac{\max_{1 \leq i \leq m} |q_{ij}|}{\sum_{k=1}^n (\max_{1 \leq i \leq m} |q_{ik}|)}. \tag{21}$$

From Eq. (10), the above multiple linear programming problem is

$$\min S = \sum_{j=1}^n W_j C_j \tag{22}$$

such that

$$\alpha_0 + \sum_{j=1}^n q_{ij} \alpha_j + (1-H)(C_0 + \sum_{j=1}^n |q_{ij}| C_j) \geq T_i - (1-H)e_i, \quad i = 1, 2, \dots, m \tag{23}$$

$$\alpha_0 + \sum_{j=1}^n q_{ij} \alpha_j - (1-H)(C_0 + \sum_{j=1}^n |q_{ij}| C_j) \leq T_i + (1-H)e_i, \quad i = 1, 2, \dots, m \tag{24}$$

$$C_j \geq 0, \quad j = 1, 2, \dots, n \tag{25}$$

From the linear programming above, the parameters can be estimated. Hence, the prediction model is

$$\tilde{T}^* = \tilde{A}_0 + q_1 \tilde{A}_1 + \dots + q_n \tilde{A}_n, \tag{26}$$

and the center of \tilde{T}_i^* is

$$T_i^* = \alpha_0 + \sum_{j=1}^n q_{ij} \alpha_j. \tag{27}$$

Based on existing advertised QoS values and the delivered quality value, the fuzzy regression line can be obtained with Eq. (26). With the obtained fuzzy regression line and new advertised QoS values $\{q_{m+1\ 1}, q_{m+1\ 2}, \dots, q_{m+1\ n}\}$, the new delivered quality value can be predicted as

$$T_{m+1}^* = \alpha_0 + \sum_{j=1}^n q_{m+1\ j} \alpha_j. \tag{28}$$

This is valuable for the decision-making of service customer prior to transactions.

5 Service Quality Trust Vector

In [6], we propose an approach of a *service quality trust vector*, which consists of *final trust level (FTL)*, *service trust trend (STT)* and *service performance consistency level (SPCL)*.

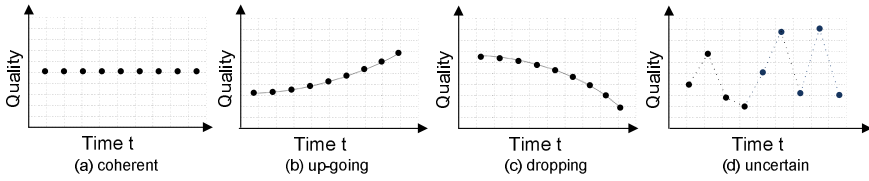


Fig. 1. Several *STT* cases

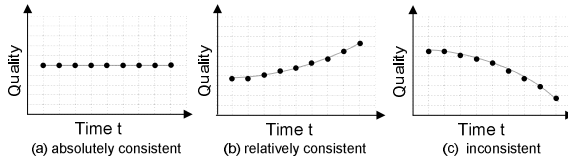


Fig. 2. Several *SPCL* cases

1. *FTL* value is provided as an indication for the global trust level of a seller or a service provider. In [6], the *FTL* value can be calculated as the *exponential moving average*:

$$T_{FTL} = \frac{\sum_{i=1}^n w_i q_i}{\sum_{i=1}^n w_i}, \tag{29}$$

where w_i is the weight for service quality q_i at time i ($i = 1 \dots n$), which can be calculated as follows:

$$w_i = \alpha^{n-i}, \quad 0 < \alpha \leq 1. \tag{30}$$

2. *STT* value shows a general trend of changes in the service quality in the near future, which is important when we choose a seller or a service provider with serious caution. Some typical cases of *STT* are depicted in Fig. 1.

In [6], the *STT* value is evaluated by introducing the *weighted least squares linear regression* method to evaluate *STT*, which is illustrated in Fig. 3. This method is used to obtain the best fit straight line from a set of given data points. This best fit straight line is characterized by the sum of weighted squared residuals having its least value, where a residual is the distance from a data points to the regression line (refer to Fig. 3). Once obtaining the regression line, its slope can be taken as our *STT* value.

3. *SPCL* value should be calculated to indicate whether the real service level is consistent over a certain period. Some typical *SPCL* cases are depicted in Fig. 2.

In [6], the *SPCL* value is calculated by

$$T_{SPCL} = 1 - 2 \frac{\sum_{i=1}^n w_i |q_i - (a_0 + a_1 t_i)|}{\sqrt{1 + a_1^2} \sum_{i=1}^n w_i}, \tag{31}$$

where a_0 and a_1 is decided by the regression line mentioned above.

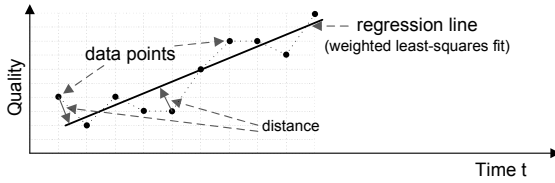


Fig. 3. Weighted least squares linear regression

In [6], only single variable is considered. However, in this paper, multiple variables are introduced, which makes a multivariable fuzzy linear regression. We should notice that it is easy to introduce the weight in Eq. (30) without complexity. Therefore, for the sake of simplicity, we omit the weight function in this paper. Based on the results in Section 4, *FTL*, *STT* and *SPCL* are redefined as follows.

Definition 3. The *FTL* value can be calculated as:

$$T_{FTL} = \frac{\sum_{i=1}^m T_i}{m}, \tag{32}$$

where T_i is the trust value at time i ($i = 1, 2, \dots, m$).

Definition 4. The *STT* value can be evaluated by the parameters of first order of Eq. (26), i.e.

$$T_{STT} = \sum_{j=1}^n \alpha_j. \tag{33}$$

Definition 5. The *SPCL* value can be evaluated by the goodness-of-fit which is defined in Eq. (10), i.e.

$$T_{SPCL} = \frac{\sum_{i=1}^m h_i}{m}, \tag{34}$$

where h_i is defined in Eq. (10).

Hence, the definition of the service quality trust vector is as follows.

Definition 6. The service quality trust vector \bar{T} consists of the *FTL* value T_{FTL} , the *STT* value T_{STT} , and the *SPCL* value T_{SPCL}

$$\bar{T} = \langle T_{FTL}, T_{STT}, T_{SPCL} \rangle, \tag{35}$$

where T_{FTL} is defined in Eq. (32), T_{STT} is decided by Eq. (33), and T_{SPCL} is defined in Eq. (34).

Moreover, with trust vectors, all service providers form a partial order set. Given two service providers P_i, P_j with service quality trust vectors $\bar{T}_i = \langle T_{FTL_i}, T_{STT_i}, T_{SPCL_i} \rangle$, and $\bar{T}_j = \langle T_{FTL_j}, T_{STT_j}, T_{SPCL_j} \rangle$ respectively, they are comparable in the following cases:

1. If $T_{STT_i} = T_{STT_j}$, $T_{SPCL_i} = T_{SPCL_j}$, and $T_{FTL_i} < T_{FTL_j}$, P_j is more preferable. We denote it as $P_j > P_i$ or $P_i < P_j$.
2. If $T_{FTL_i} = T_{FTL_j}$, $T_{SPCL_i} = T_{SPCL_j}$, and $T_{STT_i} < T_{STT_j}$, P_j is more preferable. This is denoted as $P_j > P_i$ or $P_i < P_j$.
3. If $T_{FTL_i} = T_{FTL_j}$, $T_{STT_i} = T_{STT_j}$, and $T_{SPCL_i} < T_{SPCL_j}$, P_j is more preferable. We denote it as $P_j > P_i$ or $P_i < P_j$.

In [6], a regression line is built up to indicate the service quality level, based on the time variable and the service quality value. Different from the previous work, in this paper, the fuzzy regression line is determined in multiple dimensional space, which consists of multiple independent variable axes of advertised QoS values and one dependent variable axis of delivered quality value.

6 Empirical Studies

In this section, we illustrate the results of conducted simulations to study the proposed service trust vector approach, and explain why the service trust vector is necessary and important. In addition, we explain why we adopt the fuzzy regression in this work.

6.1 Study 1

In this study, we conduct a study with six cases to illustrate why the trust vector is necessary and important. Here we only take one QoS value, which is the time, in order to illustrate the fuzzy regression method in a two dimensional figure. We set $H = 0.6$ and $e = [0.01 \ 0.01 \ \dots \ 0.01]'_{1 \times 100}$. The computed results are listed in Table 1, and the center of the regression line for each service provider in this study is also plotted in Fig. 4.

Table 1. Study 1 results

	T_{FTL}	T_{STT}	T_{SPCL}
P_1	0.6096	0.0040	0.8760
P_2	0.6092	0.0036	0.8562
P_3	0.6027	-0.0023	0.8846
P_4	0.6104	-0.0031	0.8556
P_5	0.6057	0.0005	0.8640
P_6	0.6131	-0.0008	0.8474

In each case, as plotted in Fig. 4 there is one service provider. According to Table 1, all six cases have almost the same FTL , but different T_{STT} or T_{SPCL} . Meanwhile, we can conclude that $P_1 > P_2$, $P_1 > P_3$, $P_2 > P_4$, $P_3 > P_4$, $P_5 > P_3$, $P_5 > P_6$, and $P_6 > P_4$. Namely, with solo FTL , it is not likely to depict the trust history exactly and compare service providers well.

Therefore, in this study, we can notice that the trust vector including T_{FTL} , T_{STT} and T_{SPCL} can describe the history of trust data more precisely than the solo FTL .

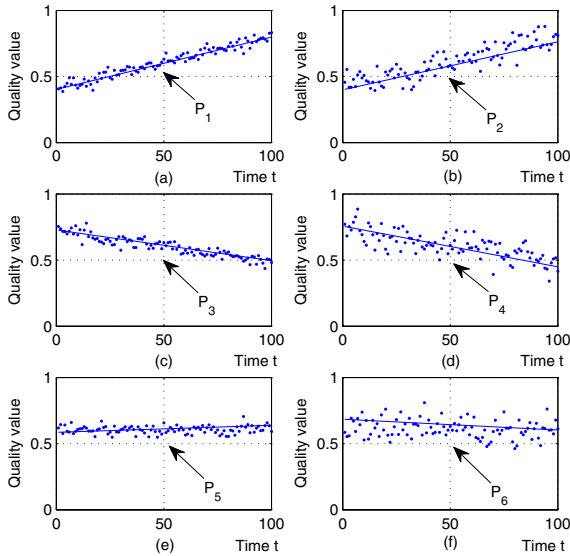


Fig. 4. Study 1

6.2 Study 2

In this study, we apply our model to predict the feedback of learner experience of teacher at Macquarie University⁴, Sydney, Australia. Table 2 illustrates an example about the feedback of learner experience⁵ of the same teacher in a course between 2005 and 2008, obtained from the Centre for Professional Development⁶ at Macquarie University. The questionnaire is designed to collect students' feedback on a teacher's teaching quality. Questions 1 to 11 are based on generic attributes of teaching quality, such as: communicated clearly, enthusiasm, good learning atmosphere, constructive feedback, etc, which can be taken as QoS values. Question 12 is "I would recommend a unit taught by this teacher to other students", which can be considered as the overall quality value. These values are also illustrated in Fig. 5.

Therefore, a fuzzy regression model can be built up to describe the relation between the input, i.e. the time and the feedback of Questions 1-11, and the output, i.e. the feedback of Question 12. In addition, the fuzzy regression model parameters are determined by the data from 2005 to 2007. Then based on the fuzzy regression model, with the input data of 2008, the corresponding output is predicted. Compared with the feedback of Question 12, the goodness-of-fit result can be obtained.

⁴ <http://www.mq.edu.au/>

⁵ http://www.mq.edu.au/learningandteachingcentre/for_staff/teaching_eval/let.htm

⁶ <http://www.cpd.mq.edu.au/>

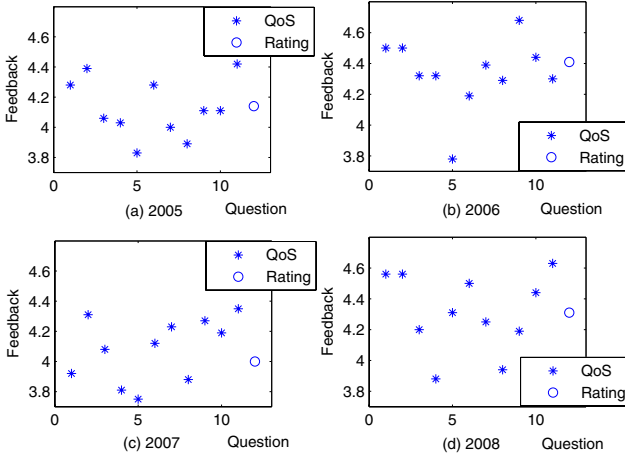


Fig. 5. Feedback of learner experience of the same teacher in a course between 2005 and 2008 from the Centre for Professional Development at Macquarie University

Table 2. Feedback of learner experience of teacher about certain course between 2005 and 2008 from Centre for Professional Development in Macquarie University

Year	QoS_1	QoS_2	QoS_3	QoS_4	QoS_5	QoS_6	QoS_7	QoS_8	QoS_9	QoS_{10}	QoS_{11}	Overall rating	Number of respondents
1	4.28	4.39	4.06	4.03	3.83	4.28	4.00	3.89	4.11	4.11	4.42	4.14	36
2	4.50	4.50	4.32	4.32	3.78	4.19	4.39	4.29	4.68	4.44	4.30	4.41	28
3	3.92	4.31	4.08	3.81	3.75	4.12	4.23	3.88	4.27	4.19	4.35	4.00	26
4	4.56	4.56	4.20	3.88	4.31	4.50	4.25	3.94	4.19	4.44	4.63	4.31	16

Let $H = 0.6$ and $e_i = 0.1$, and Eq. (22) becomes

$$S = 0.01962C_0 + 0.05886C_1 + 0.08829C_2 + 0.08829C_3 + 0.08476C_4 \tag{36}$$

$$+ 0.08476C_5 + 0.07514C_6 + 0.08397C_7 + 0.08613C_8 + 0.08417C_9 \tag{37}$$

$$+ 0.09182C_{10} + 0.08711C_{11} + 0.08672C_{12} \tag{38}$$

From the linear programming, the fuzzy regression model is

$$\tilde{T}_i^* = \tilde{A}_0 + q_{i1}\tilde{A}_1 + \dots + q_{in}\tilde{A}_n, \tag{39}$$

where

$$\mu_{\tilde{A}_0}(x) = \begin{cases} 1, & x = -0.1036, \\ 0, & x \neq -0.1036, \end{cases} \tag{40}$$

$$\mu_{\tilde{A}_1}(x) = \begin{cases} 1, & x = -0.03569, \\ 0, & x \neq -0.03569, \end{cases} \tag{41}$$

$$\mu_{\tilde{A}_2}(x) = \begin{cases} 1, & x = 0.2128, \\ 0, & x \neq 0.2128, \end{cases} \tag{42}$$

$$\mu_{\tilde{A}_3}(x) = \begin{cases} 1, & x = -0.1469, \\ 0, & x \neq -0.1469, \end{cases} \tag{43}$$

$$\mu_{\tilde{A}_4}(x) = \begin{cases} 1, & x = 0.07865, \\ 0, & x \neq 0.07865, \end{cases} \tag{44}$$

$$\mu_{\tilde{A}_5}(x) = \begin{cases} 1, & x = 0.2084, \\ 0, & x \neq 0.2084, \end{cases} \tag{45}$$

$$\mu_{\tilde{A}_6}(x) = \begin{cases} 1, & x = -0.1315, \\ 0, & x \neq -0.1315, \end{cases} \tag{46}$$

$$\mu_{\tilde{A}_7}(x) = \begin{cases} 1, & x = -0.06863, \\ 0, & x \neq -0.06863, \end{cases} \tag{47}$$

$$\mu_{\tilde{A}_8}(x) = \begin{cases} 1, & x = 0.1396, \\ 0, & x \neq 0.1396, \end{cases} \tag{48}$$

$$\mu_{\tilde{A}_9}(x) = \begin{cases} 1, & x = 0.2033, \\ 0, & x \neq 0.2033, \end{cases} \tag{49}$$

$$\mu_{\tilde{A}_{10}}(x) = \begin{cases} 1, & x = 0.01623, \\ 0, & x \neq 0.01623, \end{cases} \tag{50}$$

$$\mu_{\tilde{A}_{11}}(x) = \begin{cases} 1, & x = 0.1504, \\ 0, & x \neq 0.1504, \end{cases} \tag{51}$$

$$\mu_{\tilde{A}_{12}}(x) = \begin{cases} 1, & x = 0.3659, \\ 0, & x \neq 0.3659, \end{cases} \tag{52}$$

Table 3. Goodness-of-fit results in Study 2

Year	Original rating	Prediction rating	Error percentage
2008	4.31	4.1418	3.9030%

From Eq. (39), the corresponding goodness-of-fit results are listed in Table 3 with error percentage of 3.9030%. Obviously, we can see the fuzzy regression model predicts well. With the data in more years, the prediction will become more accurate.

7 Conclusions

In this paper, we propose a fuzzy regression based trust vector approach to service-oriented applications, which includes *final trust level (FTL)*, *service trust trend (STT)*, and *service performance consistency level (SPCL)*. From our analytical and empirical studies, we can see that the proposed approach can depict trust history exactly. Meanwhile, with the advertised QoS values, the delivered quality value can be predicted based on our model. It offers more information to service customers for their decision-making in the selection of trustworthy service providers.

References

1. Damiani, E., di Vimercati, S.D.C., Paraboschi, S., Samarati, P., Violante, F.: A reputation based approach for choosing reliable resources in peertopeer networks. In: Proceedings of ACM CCS 2002, Washington, DC, USA, November 2002, pp. 207–216 (2002)
2. Hines, W.W., Montgomery, D.C., Goldsman, D.M., Borror, C.M.: Probability and Statistics in Engineering. John Wiley & Sons, Inc., Chichester (2003)
3. Jang, J.-S.R., Sun, C.-T., Mizutani, E.: Neuro-Fuzzy and Soft Computing. Prentice Hall, Englewood Cliffs (1997)
4. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International WWW Conference, Budapest, Hungary (May 2003)
5. Kosko, B.: Fuzzy Engineering. Prentice Hall, Englewood Cliffs (1997)
6. Li, L., Wang, Y.: A trust vector approach to service-oriented applications. In: Proceedings of The IEEE International Conference on Web Services (ICWS 2008), pp. 270–277 (2008)
7. Lin, K.-J., Lu, H., Yu, T., en Tai, C.: A reputation and trust management broker framework for web applications. In: Proceedings of The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE 2005), March 2005, pp. 262–269 (2005)
8. Marti, S., Garcia-Molina, H.: Limited reputation sharing in P2P systems. In: Proceedings of ACM EC 2004, New York, USA, May 2004, pp. 91–101 (2004)
9. Song, S., Hwang, K., Zhou, R., Kwok, Y.-K.: Trusted P2P transactions with fuzzy reputation aggregation. IEEE Internet Computing 9(6), 24–34 (2005)
10. Vu, L.-H., Hauswirth, M., Aberer, K.: QoS-based service selection and ranking with trust and reputation management. In: Proceedings of 13th International Conference on Cooperative Information Systems (CoopIS 2005), October 31–November 4 (2005)
11. Wang, Y., Lin, K.-J., Wong, D.S., Varadharajan, V.: The design of a rule-based and event-driven trust management framework. In: The IEEE International Conference on e-Business Engineering (ICEBE 2007), Hong Kong, October 2007, pp. 97–104 (2007)
12. Wang, Y., Varadharajan, V.: Interaction trust evaluation in decentralized environments. In: Bauknecht, K., Bichler, M., Pröll, B. (eds.) EC-Web 2004. LNCS, vol. 3182, pp. 144–153. Springer, Heidelberg (2004)
13. Wang, Y., Varadharajan, V.: *Trust²*: Developing trust in peer-to-peer environments. In: Proceedings of 2005 IEEE International Conference on Services Computing (SCC 2005), Orlando, Florida, USA, July 2005, pp. 24–31 (2005)
14. Xiong, L., Liu, L.: PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans. on Knowledge and Data Engineering 16(7), 843–857 (2004)
15. Yu, B., Singh, M.P., Sycara, K.: Developing trust in large-scale peer-to-peer systems. In: Proceedings of 2004 IEEE First Symposium on Multi-Agent Security and Survivability, August 2004, pp. 1–10 (2004)
16. Zacharia, G., Maes, P.: Trust management through reputation mechanisms. Applied Artificial Intelligence Journal 9, 881–908 (2000)

Theories of Trust for Communication Protocols

Ji Ma^{1,2}, Mehmet A. Orgun¹, and Abdul Sattar²

¹ Department of Computing, Macquarie University, Sydney, NSW 2109, Australia

² School of Information and Communication Technology, Nathan campus, Griffith University, Queensland 4111, Australia

Abstract. Trust is a critical issue for communication protocols in open systems that operate in dynamic and highly uncertain environments. It influences not only the specification of security policies but also the techniques needed to manage and implement security policies. A formal system for the specification of trust for such systems requires the ability to reason about agent beliefs as well as the evolution of the system through time. In this paper, we use a fibred logic called FL which is obtained by combining a belief logic with a temporal logic for specifying agent beliefs and establishing trust theories for communication protocols. A theory of trust for a given system is captured by a set of rules in FL that describes the trust of agents in the system. This enables automated reasoning about theories of trust using the decision procedures of FL such as axiom systems and tableaux. Theories of trust are generally established based on the initial trust of agents in the security mechanisms of the system in which they are deployed. Such theories provide a foundation for reasoning about agent beliefs as well as security properties that systems may satisfy.

1 Introduction

Trust is a critical issue for communication protocols in open systems that operate in dynamic and highly uncertain environments [1]. It influences not only the specification of security policies but also the techniques needed to manage and implement security policies. A communication protocol is a set of standard rules specifying data representation, signalling, authentication and error detection required to send messages between agents over a communication channel. After authentication, two agents should be entitled to believe that they are communicating with each other and not with intruders. So it is important to express such beliefs precisely and to capture the reasoning that leads to them [2].

In a trust model proposed by Liu *et al.* [3], it is assumed that for security considerations initially agents may not trust any one but the security mechanisms (as special agents) of a system whose trustworthiness has been verified based on required evaluation criteria. Thus, the beliefs of agents can be obtained based on their initial trust in the security mechanisms of the system. The initial trust of agents in the system can be encapsulated in a notion of trust and represented as a set of rules in a chosen logical framework. These rules together with those of the logic form a theory of trust for the system [4]. Such theories provide a foundation for reasoning about agent beliefs as well as security properties that systems may satisfy. We assume that a communication protocol leads to such a trust theory.

There are many studies discussing how to use a kind of belief logic to describe communication systems. Burrows *et al.* [2] proposed a logic called BAN to describe the beliefs of trustworthy parties involved in authentication protocols and the evolution of these beliefs. Many researchers have developed extensions of the BAN logic, such as GNY logic [5], VO logic [6], SVO logic [7], and AT logic [8]. Many general purpose logics have also been applied for the analysis of authentication [9,10,11,12]. In more recent works on the investigation of agent beliefs, Bacchus *et al.* [13] discussed knowledge bases to degrees of belief; Gabbay *et al.* [14] proposed an algorithmic approach for belief revision; Weydert [15] proposed a ranking model of belief revision in rational agents; Smets and Ristic [16] proposed a belief function for joint tracking and classification. In our earlier work [17], we have applied a temporalised belief logic called TML⁺ to the specification of authentication protocols. Since TML⁺ is obtained using temporalisation [18], it only allows reasoning about the temporal aspects of beliefs, not about beliefs about temporal aspects.

In this paper, we instead use a fibred logic called FL [19,20,21] to establish trust theories for communication protocols. FL is obtained by combining a belief logic called TML (Typed Modal Logic) [4] with a temporal logic called SLTL (Simple Linear time Temporal Logic) [22] using a more powerful technique called fibring [23]. This logic allows us to express agent beliefs as temporal propositions that may vary through time as well as beliefs about temporal propositions. With the logic and theories, we are able to specify, reason about and verify communication protocols for agent-based systems operating in dynamic environments. In this paper, we first give an introduction to the fibred logic for completeness, further discuss a method for establishing generic trust theories for communication protocols, and apply it to three well-known protocols.

The rest of this paper is organised as follows: Section 2 discusses agent beliefs and introduces the fibred logic FL. Section 3 present a generic method for establishing trust theories for communication protocols, and gives three examples. Section 4 discusses formal reasoning methods for communication protocols based on trust theories. Section 5 concludes the paper with a brief summary and future research directions.

2 An Introduction to the Fibred Logic

Combining logics is an active research area in modelling agent systems. Some major combination methods include: **fusion** [24], **fibring** [23], **products** [25], **temporalisation** [18], **hierarchy combination** [26], and **free mixture combination** [27]. Different combination methods naturally produce quite different combined logics with different expressive capabilities.

For reasoning about agent beliefs, we may need to start with a belief logic specifically defined for this purpose. Liu [4] proposed a belief logic, called TML, which extends first-order logic with typed variables and belief operators. However, this logic lacks a temporal dimension, and therefore it may not be able to express dynamic changes of systems and their properties. In a later work, Liu *et al.* [19] used the fibring technique proposed by Gabbay [28] to add a temporal dimension to TML, by combining it with the Simple Linear-time Temporal logic (SLTL). The resulting logic, FL, can be applied for reasoning about time-dependent properties regarding agent beliefs in secure systems.

FL has been applied to the analysis of stream authentication protocols in which time plays a central role [20,29,21].

The fibred logic (FL for short) has two classes of modal operators: (1) belief operators; and (2) temporal operators. The belief operator, \mathbf{B}_a , is intended to denote “agent a believes that”. The belief operators are those of TML whereas the temporal operators are those of SLTL. SLTL is a linear-time logic where the collection of time points is the set of natural numbers with a usual ordering relation $<$. It has two temporal operators, **first** and **next**, which refer to the initial moment and the next moment in time respectively [22]. The meaning of SLTL formulas are defined with respect to given local clocks (subsequences of a global clock). The global clock is the increasing sequence of natural numbers, i.e., $(0, 1, 2, \dots)$ and a local clock (or simply, a clock) is defined as an infinite subsequence of the global clock. SLTL allows reasoning about events with varying rates of progress.

Table 1 gives an intuitive explanation of the interpretation of the temporal operators of SLTL [17].

Table 1. Interpretation of temporal operators

Formula	Truth value								
ϕ	T	T	F	F	T	F	T	F	...
first ϕ	T	T	T	T	T	T	T	T	...
next ϕ	T	F	F	T	F	T	F
Time	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	...

In Table 1, ϕ is a logical formula, T represents value **true** and F value **false**, and c is a given local clock where each t_i is a moment on the clock.

A time model for the logic SLTL has the form $\mathbf{c} = \langle C, <, \pi^{(\mathbf{c})} \rangle$, where $C = (t_0, t_1, t_2, \dots)$ is a clock, $<$ is the usual ordering relation over C , and $\pi^{(\mathbf{c})}$ is an assignment function that gives a value $\pi^{(\mathbf{c})}(t, p) \in \{true, false\}$ for any any time point t in C and any atomic formula p .

Then the semantics of the temporal operators of SLTL are given as follows:

- $\mathbf{c}, t_i \models \mathbf{first} \phi$ iff $t_0 \models \phi$.
- $\mathbf{c}, t_i \models \mathbf{next} \phi$ iff $t_{i+1} \models \phi$.
- satisfaction in the model $\mathbf{c} = \langle C, <, \pi^{(\mathbf{c})} \rangle$ is defined as satisfaction at some time point on C .

Let us assume that there are n agents a_1, \dots, a_n and there are n corresponding modal operators $\mathbf{B}_{a_1}, \dots, \mathbf{B}_{a_n}$ in the logic, where \mathbf{B}_{a_i} ($1 \leq i \leq n$) stands for “agent a_i believes that”. A classical Kripke model [30] for TML is defined as a tuple $\mathbf{m} = \langle S, R_1, \dots, R_n, \pi \rangle$, where S is the set of states or possible worlds; and each R_i , $i = 1, \dots, n$, is a possibility relation over S , according to agent a_i , and is defined as follows: R_i is a non-empty set consisting of state pairs (s, t) such that $(s, t) \in R_i$ iff, at state s , agent a_i considers the state t possible; and π is the assignment function, which gives a value $\pi(s, p) \in \{true, false\}$ for any $s \in S$ and atomic formula p . Formula ϕ is satisfiable in the model \mathbf{m} if there exists $s \in S$ such that $\mathbf{m}, s \models \phi$.

The semantics of the belief operators of TML is given as follows:

- $\mathcal{M}_{tml}, w \models \mathbf{B}_{a_i} \phi$, iff for all w_j , $(w, w_j) \in R_i$, $\mathcal{M}_{tml}, w_j \models \phi$.

In our previous works [26,31,17], we used a temporal belief logic called TML^+ to formalise agent systems. However, there are certain restrictions on the use of temporal and belief operators, because of the hierarchical combination of belief and temporal logics used. There the temporal logics SLTL is added onto the belief logic TML in such a way that temporal operators can never be within the scope of a belief operator in TML^+ . Hence in TML^+ , we cannot express a statement asserting that some agent believes an event to happen at some time, e.g., we can have the formula **first** $\mathbf{B}_{Alice} \phi$, but can not have the formula \mathbf{B}_{Alice} **first** ϕ . The logic FL is obtained through the use of fibring technique [28] for combining the logic TML with the logic SLTL, which treats temporal operators and belief operators equally.

In FL, both the assertions **first** $\mathbf{B}_{Alice} \phi$ and \mathbf{B}_{Alice} **first** ϕ are legal formulas. FL has stronger expressive power than TML^+ , since these operators can occur in any order in formulas, e.g., we may have **next** \mathbf{B}_{Alice} **first** **next** ϕ , which means that “at the next moment Alice believes that at the next moment after initial time ϕ is true.”

Fibring also allows transfer of certain properties of the constituent logics to the resulting logic. For instance, the soundness for the logic FL depends on the soundness theorems for belief logic and SLTL, and is not difficult to prove. For more details on the completeness of the logic FL we refer the reader to the literature [23,19].

3 Establishing Theories of Trust for Communication Protocols

The components of a communications system serve a common purpose, basically following certain rules so that the system works properly. In this section we outline a generic method for establishing trust theories for communication systems, and give three examples to show how to construct a trust theory for a given communication protocol. All the protocol specifications used in this paper are available from [32].

3.1 Establishing Trust Theories for Communication Systems

A trust theory for a given authentication system consists of a set of rules which can be used for reasoning about agent beliefs and security properties that the system may satisfy [31]. Establishing a trust theory for a given authentication system involves the following steps:

1. Analysing how the communication system works.
2. Analysing security mechanisms of the system, and identifying agents in it and initial security assumptions.
3. Defining appropriate predicates to express the main properties of the system
4. Defining the rules that describe the functions and behaviours of the system.

For specifying communication protocols, we define the following types:

Agents: A, B, S, I
Messages: M_1, M_2, M_3, \dots
Nonces: N_a, N_b, N_s, \dots
Keys: $K_{as}, K_{ab}, K_{bs}, \dots$
Clocks: ck_s, ck_a, ck_b, \dots

Here A, B, and S to denote agents, I denotes intruder; M_i denote a message; $N_a, N_b,$ and N_s denote specific nonces; $k_{ab}, k_{as},$ and k_{bs} denote specific keys; and $ck_a, ck_b,$ and ck_s denote specific clocks.

In order to establish a theory of trust for authentication systems, we define the following standard predicates:

- $send(a, b, msg)$: Agent a sends a message msg to another agent b .
- $receive(a, msg)$: Agent a receives a message msg .
- $secure(k)$: Key k is secure.
- $fresh(t)$: Timestamp t is fresh.
- $Duplicated(t)$: Timestamp t is duplicated.
- $reliable(msg)$: Message msg is reliable.
- $reject(msg)$: Message msg is rejected.
- $synchronized(ck_1, ck_2)$: Clocks ck_1 and ck_2 are synchronized.

Now we discuss the standard communication axioms for authentication systems. In the following axioms, all variables are assumed to be universally quantified. The axioms are valid for any given pairs of agents a and b .

- R1. $\mathbf{B}_a secure(k) \wedge receive(a, \{X\}k) \leftrightarrow \mathbf{B}_a reliable(X)$.
R2. $\mathbf{B}_a secure(k) \wedge receive(a, \{X[T]\}k) \wedge \mathbf{B}_a fresh(T) \leftrightarrow \mathbf{B}_a reliable(X)$.
R3. $receive(a, \{X[T]\}k) \wedge \mathbf{B}_a duplicated(T) \leftrightarrow reject(X)$.
R4. $send(a, b, X) \leftrightarrow receive(b, X)$.
R5. $receive(a, (X, Y)) \leftrightarrow receive(a, X) \wedge receive(a, Y)$.
R6. $\mathbf{B}_a fresh((X, Y)) \leftrightarrow \mathbf{B}_a fresh(X) \wedge \mathbf{B}_a fresh(Y)$.
R7. $\mathbf{B}_a reliable((X, Y)) \leftrightarrow \mathbf{B}_a reliable(X) \wedge \mathbf{B}_a reliable(Y)$.

Here $\{X\}k$ means that message X is encrypted with key k . The meaning of these axioms are defined as follows: Rule R1 says that, agent a believes that encryption key k is secure, and receives a message X encrypted with key k , iff agent a believes that the message X is reliable. Rule R2 says that, agent a believes that encryption key k is secure, and receives a message X which contains a timestamp T (denoted by $X[T]$), also the message is encrypted with key k , and agent a believes the timestamp is fresh, iff agent a believes that the message is reliable. Rule R3 says that, agent a receives a message X which contains a timestamp T , and agent a believes that the timestamp is duplicated, iff the message will be rejected. Rule R4 assumes that sending and receiving a message happen simultaneously. The meanings of Rule R5 - R7 are straightforward.

The security properties of this protocol are based on the confidentiality of encryption keys and synchronized clocks.

- Encryption keys: If an intruder does not know those encryption keys used encrypt messages, in other words, if encryption keys are not compromised, then the intruder cannot learn those messages.

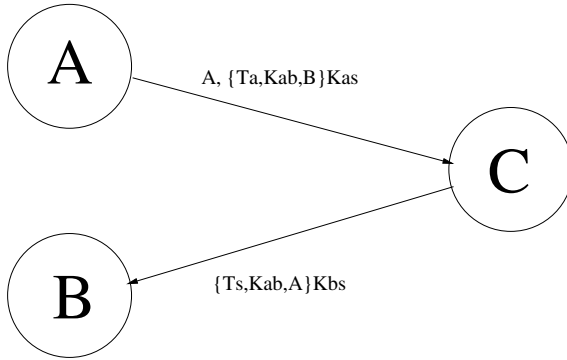


Fig. 1. Wide Mouthed Frog protocol

- Synchronized clocks: Synchronized clocks are used to guarantee the correctness of timestamps.

In the following, we give three examples to show how to establish a specific trust theory for a given protocol.

3.2 Wide Mouthed Frog Protocol

The Wide Mouthed Frog Protocol is a symmetric key management protocols involving a trusted third party. The protocol can be specified as follows in the security protocol notation: where an agent A is authenticating itself to another agent B using a server S [2][33][34]. The authentication process is as follows (see Figure 2):

1. $A \rightarrow S: A, \{T_a, K_{ab}, B\}_{K_{as}}$
2. $S \rightarrow B: \{T_s, K_{ab}, A\}_{K_{bs}}$

This is a multi-agent communication system. The system uses such security mechanisms to keep the communication secure. This implies that principals would trust the security mechanisms of this system. Initially agents would trust that:

- Authentication agent, that is, the server (S) is trustworthy.
- All encryption keys are secure.
- The server's clock and the clients' clocks are synchronized.

That is, principles involving in the system must have an initial trust in the set of security mechanisms, denoted by \mathcal{M} ,

$$\mathcal{M}_w = \{S, k_{ab}, k_{as}, k_{bs}, ck_s, ck_a, ck_b\}.$$

And we give the following assumptions:

$$\begin{array}{ll} \mathbf{B}_a \text{ secure}(k_{ab}) & \mathbf{B}_a \text{ secure}(k_{as}) \\ \mathbf{B}_s \text{ secure}(k_{as}) & \mathbf{B}_s \text{ secure}(k_{bs}) \end{array}$$

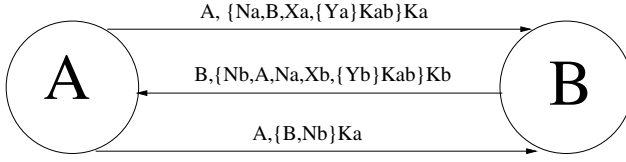


Fig. 2. BAN modified version of CCITT X.509(3) Protocol

$\mathbf{B}_b \text{ secure}(k_{bs})$	$\mathbf{B}_b \text{ secure}(k_{ab})$
$\mathbf{B}_s \text{ synchronized}(ck_s, ck_a)$	$\mathbf{B}_s \text{ synchronized}(ck_s, ck_b)$
$\mathbf{B}_a \text{ synchronized}(ck_s, ck_a)$	$\mathbf{B}_b \text{ synchronized}(ck_s, ck_b)$

We have the following rule that describe the authentication procedure.

$$W1. \mathbf{B}_s \text{ reliable}(\{A, \{T_a, K_{ab}, B\}_{K_{a,s}}\}) \leftrightarrow \text{next send}(S, B, \{T_s, K_{ab}, A\}_{K_{bs}}).$$

Now we have established a theory $T_w = \{R1, R12, R3, R4, R5, R6, R7, W1\}$ for Wide Mouthed Frog protocol. Since agents trust the security mechanisms of the protocol, the trust theory can therefore be used for reasoning about agent beliefs about the system.

3.3 BAN Modified Version of CCITT X.509(3) Protocol

The protocol is modified version of CCITT X.509(3). Compared to CCITT X.509(3), the identity of B has been added to the signature in message 3. This prevents the attack on the CCITT X.509 (3) protocol, which can occur when B does not check the timestamps [2]. The authentication process is as follows, (see Figure 3):

1. $A \rightarrow B: A, \{N_a, B, X_a, \{Y_a\}_{K_{ab}}\}_{K_a}$
2. $B \rightarrow A: B, \{N_b, A, N_a, X_b, \{Y_b\}_{K_{ab}}\}_{K_b}$
3. $A \rightarrow B: A, \{B, N_b\}_{K_a}$

Here X_a and X_b standard for specific user data. Initially agents within the system would trust that:

- All encryption keys are secure.
- The clients' clocks are synchronized.
- The clients provide correct user data.

That is, principles involving in the system must have an initial trust in the set of security mechanisms, which is,

$$\mathcal{M}_c = \{k_{ab}, k_a, k_b, ck_a, ck_b\}.$$

And we give the following assumptions:

$\mathbf{B}_a \text{ secure}(k_{ab})$	$\mathbf{B}_a \text{ secure}(k_a)$
$\mathbf{B}_a \text{ secure}(k_b)$	$\mathbf{B}_b \text{ secure}(k_{ab})$
$\mathbf{B}_b \text{ secure}(k_a)$	$\mathbf{B}_b \text{ secure}(k_b)$
$\mathbf{B}_a \text{ synchronized}(ck_a, ck_b)$	$\mathbf{B}_b \text{ synchronized}(ck_a, ck_b)$

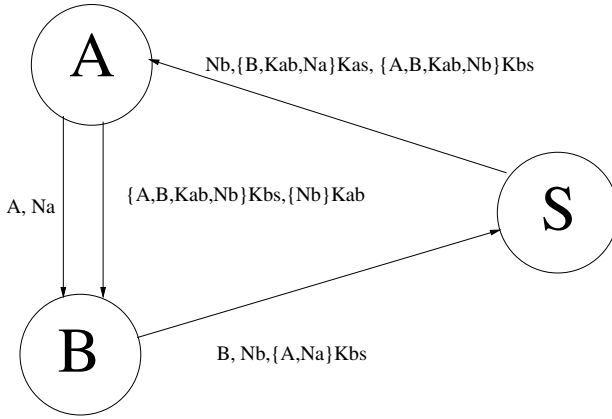


Fig. 3. Paulson's strengthened version of Yahalom Protocol

We have the following rules that describe the authentication procedure.

- C1. $\mathbf{B}_b \text{ reliable}(\{A, \{N_a, B, X_a, \{Y_a\}K_{ab}\}K_a\}) \leftrightarrow$
 next send($B, A, \{B, \{N_b, A, N_a, X_b, \{Y_b\}K_{ab}\}K_b\}$).
- C2. $\mathbf{B}_a \text{ reliable}(\{B, \{N_b, A, N_a, X_b, \{Y_b\}K_{ab}\}K_b\}) \leftrightarrow$
 next send($A, B, \{A, \{B, N_b\}K_a\}$).

Now we have established a theory $T_c = \{R1, R12, R3, R4, R5, R6, R7, C1, C2\}$ for BAN modified version of CCITT X.509(3) Protocol.

3.4 Paulson's Strengthened Version of Yahalom Protocol

Yahalom protocol distributes fresh symmetric shared keys by trusted servers and mutual authentication. For preventing A to reuse an old key K_{ab} , the nonce N_b is added to the second cipher sent by S in the strengthened version. The authentication process is as follows, (see Figure 4):

1. $A \rightarrow B: A, N_a$
2. $B \rightarrow S: B, N_b, \{A, N_a\}K_{bs}$
3. $S \rightarrow A: N_b, \{B, K_{ab}, N_a\}K_{as}, \{A, B, K_{ab}, N_b\}K_{bs}$
3. $A \rightarrow B: \{A, B, K_{ab}, N_b\}K_{bs}, \{N_b\}K_{ab}$

Initially agents within the system would trust that:

- Authentication agent, that is, the server (S) is trustworthy.
- All encryption keys are secure.
- The server's clock and the clients' clocks are synchronized.

That is, principles involving in the system must have an initial trust in the set of security mechanisms, which is,

$$\mathcal{M}_y = \{S, k_{ab}, k_{as}, k_{bs}, ck_s, ck_a, ck_b\}$$

. And we give the following assumptions:

$\mathbf{B}_a \text{ secure}(k_{ab})$	$\mathbf{B}_a \text{ secure}(k_{as})$
$\mathbf{B}_a \text{ secure}(k_{bs})$	$\mathbf{B}_b \text{ secure}(k_{ab})$
$\mathbf{B}_b \text{ secure}(k_{as})$	$\mathbf{B}_b \text{ secure}(k_{bs})$
$\mathbf{B}_s \text{ secure}(k_{as})$	$\mathbf{B}_s \text{ secure}(k_{bs})$
$\mathbf{B}_s \text{ synchronized}(ck_s, ck_a)$	$\mathbf{B}_s \text{ synchronized}(ck_s, ck_b)$
$\mathbf{B}_a \text{ synchronized}(ck_s, ck_a)$	$\mathbf{B}_b \text{ synchronized}(ck_s, ck_b)$

We have the following rules that describe the authentication procedure.

- Y1. $\mathbf{B}_b \text{ reliable}(A, N_a) \leftrightarrow \text{next send}(B, S, \{B, N_b, \{A, N_a\}K_{bs}\})$.
Y2. $\mathbf{B}_s \text{ reliable}(\{B, N_b, \{A, N_a\}K_{bs}\}) \leftrightarrow$
 $\text{next send}(S, A, \{N_b, \{B, K_{ab}, N_a\}K_{as}, \{A, K_{ab}, N_b\}K_{bs}\})$.
Y3. $\mathbf{B}_a \text{ reliable}(\{N_b, \{B, K_{ab}, N_a\}K_{as}, \{A, B, K_{ab}, N_b\}K_{bs}\}) \leftrightarrow$
 $\text{next send}(A, B, \{\{A, B, K_{ab}, N_b\}K_{bs}, \{N_b\}K_{ab}\})$.

Now we have established a theory $T_c = \{R1, R12, R3, R4, R5, R6, R7, Y1, Y2, Y3\}$ for Paulson's strengthened version of Yahalom Protocol.

4 Formal Reasoning

Since we have established trust theories for communication protocols, we are able to reason about the security properties of those protocols. The security properties of communication protocols are based on the confidentiality of encryption keys and synchronized clocks. The reasoning process is based on axioms and inference rules.

In the following, we discuss the axioms and rules of inference of FL that may be used in reasoning about trust theories for authentication protocols in agent-based systems. The axioms and rules of inference of the combined logic FL come from the two constituent logics (TML and SLTL) in a systematic way; we refer the reader to the literature for further details, see [20,21].

The axiom set of FL consists of the following axiom schemata. Let ∇ stand for any modal operator.

- A0. all axioms of classical first-order logic, including substitution.
A1. $\nabla(\varphi \rightarrow \psi) \leftrightarrow (\nabla\varphi \rightarrow \nabla\psi)$.
A2. $\nabla(\varphi \wedge \psi) \leftrightarrow (\nabla\varphi) \wedge (\nabla\psi)$.
A3. $\forall X(\nabla\varphi(X)) \rightarrow \nabla(\forall X\varphi(X))$.
A4. $\mathbf{B}_i(\neg\varphi) \rightarrow \neg(\mathbf{B}_i\varphi)$, ($1 \leq i \leq n$).
A5. $\mathbf{first}(\neg\varphi) \leftrightarrow \neg(\mathbf{first}\varphi)$.
A6. $\mathbf{next}(\neg\varphi) \leftrightarrow \neg(\mathbf{next}\varphi)$.
A7. $\mathbf{first}(\mathbf{first}\varphi) \leftrightarrow \mathbf{first}\varphi$.
A8. $\mathbf{next}(\mathbf{first}\varphi) \leftrightarrow \mathbf{first}\varphi$.

The inference rules of the logic FL include:

- IR1. $\frac{\varphi, \varphi \rightarrow \psi}{\psi}$ (Modus Ponens)
IR2. $\frac{\forall X\varphi(X)}{\varphi(Y)}$ (Instantiation)

$$\text{IR3. } \frac{\varphi(X)}{\forall X \varphi(X)} \quad (\text{Generalisation})$$

$$\text{IR4. } \frac{\varphi}{\mathbf{first} \varphi}, \frac{\varphi}{\mathbf{next} \varphi} \quad (\text{Temporal Necessitation})$$

$$\text{IR5. } \frac{\varphi}{\mathbf{B}_i \varphi} \quad (\text{Belief Necessitation})$$

$$\text{IR6. } \frac{\varphi, \psi}{\varphi \wedge \psi} \quad (\wedge_introduction)$$

$$\text{IR7. } \frac{\varphi \wedge \psi}{\varphi}, \frac{\varphi \wedge \psi}{\psi} \quad (\wedge_elimination)$$

$$\text{IR8. } \frac{\varphi, \varphi \leftrightarrow \psi}{\psi}, \frac{\psi, \varphi \leftrightarrow \psi}{\varphi} \quad (\leftrightarrow_elimination)$$

Logical consequence is the relation that holds between a set of formulas and a formula, we write:

$$G \models_{FL} U \Rightarrow \phi,$$

which denotes that formula ϕ is derived from G and U , G is the set of global assumptions, and U is the set of local assumptions. ϕ is a logical consequence that follows from G and U .

For reasoning about communication systems, we have a trust theory T and G as global assumption set. To show a security property (a formula) is valid in this theory, we need to prove it is a logical consequence of $T \cup G$. That is, we need to prove that:

$$T \cup G \models_{FL} U \Rightarrow \phi.$$

We give a proof example as follows:

Assuming that we have the following global assumption.

$$(a) \mathbf{B}_a \text{ secure}(k_{a,b})$$

and the following local assumptions.

$$(b) \mathbf{first next} \text{ receive}(a, \{msg\}k_{a,b}) \\ \mathbf{B}_a \text{ secure}(k_{a,b}) \wedge \text{receive}(a, \{msg\}k_{a,b}) \rightarrow \mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b})$$

Then, we can prove that

$$(c) \mathbf{first next} (\mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b}))$$

That is,

$$G \models_{FL} U \Rightarrow \mathbf{first next} (\mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b}))$$

The proof is given as follows:

- (1) $\mathbf{B}_a \text{ secure}(k_{a,b})$ (assumption)
- (2) $\mathbf{next} \mathbf{B}_a \text{ secure}(k_{a,b})$ (from (1), by rule IR4)
- (3) $\mathbf{first} \mathbf{next} \mathbf{B}_a \text{ secure}(k_{a,b})$ (from (2), by rule IR4)
- (4) $\mathbf{first} \mathbf{next} \text{ receive}(a, \{msg\}k_{a,b})$ (assumption)
- (5) $\mathbf{first} \mathbf{next} \mathbf{B}_a \text{ secure}(k_{a,b}) \wedge \mathbf{first} \mathbf{next} \text{ receive}(a, \{msg\}k_{a,b})$
(from (3) & (4), by IR6)
- (6) $\mathbf{first} \mathbf{next} (\mathbf{B}_a \text{ secure}(k_{a,b}) \wedge \text{ receive}(a, \{msg\}k_{a,b}))$ (from (5), by axiom A2)
- (7) $\mathbf{B}_a \text{ secure}(k_{a,b}) \wedge \text{ receive}(a, \{msg\}k_{a,b}) \rightarrow \mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b})$ (assumption)
- (8) $\mathbf{first} \mathbf{next} (\mathbf{B}_a \text{ secure}(k_{a,b}) \wedge \text{ receive}(a, \{msg\}k_{a,b})) \rightarrow$
 $\mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b})$ (from (7), by repeated rule IR4)
- (9) $\mathbf{first} \mathbf{next} (\mathbf{B}_s \text{ secure}(k_{a,b}) \wedge \text{ receive}(a, \{msg\}k_{a,b})) \rightarrow$
 $\mathbf{first} \mathbf{next} \mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b})$ (from (8), by axiom A1)
- (10) $\mathbf{first} \mathbf{next} \mathbf{B}_a \text{ send}(b, a, \{msg\}k_{a,b})$
(from (6) & (9), by rule IR1)

□

The last formula is what we want to show. This proof process involves the use of axioms and inference rules of FL, including those rules on temporal operators as well as Modus Ponens (IR1).

The security properties of communication systems can be proved using either axiomatic proof system, tableaux proof system or model checking. For more details on the analysis of communication protocols such as Kerberos [35], we refer the reader to our previous work [17|21].

5 Conclusion

We have proposed a formal approach based on the use of a fibred belief logic to specifying agent beliefs and establishing trust theories for communication protocols. Our approach is very general, it could be useful in the designing, implementing and verifying security policies for communication systems.

As we pointed before, there are a number of logics which have been developed for specify, and reasoning about agents beliefs, especially BAN Logic family have widely been discussed and applied for the analysis of authentication protocols. Comparing with BAN logic, we combine a belief logic with a temporal logic using fibring in a systematic way, and use the inference rules of the logic. Therefore our approach is very flexible and has a stronger expressive power.

Future works include investigating reasoning techniques for agent beliefs in real life applications, mechanizing verification for communication systems, and the development of a practical tool. We also plan to develop generic methods for establishing trust theories for a broader class of security policies and protocols.

Acknowledgements

This research has been supported in part under Australian Research Council's Discovery Projects funding scheme (project number DP0452628).

References

1. Blaze, M., Feigenbaum, J., Lacy, J.: Decentralized trust management. In: Proceedings of the 1996 IEEE Computer Society Symposium on research in Security and Privacy, pp. 164–173 (1996)
2. Burrows, M., Abadi, M., Needham, R.M.: A logic of authentication. *ACM Transactions on Computer Systems* 8(1), 18–36 (1990)
3. Liu, C., Ozols, M.A.: Trust in secure communication systems – the concept, representations, and reasoning techniques. In: McKay, B., Slaney, J.K. (eds.) *Canadian AI 2002. LNCS*, vol. 2557, pp. 60–70. Springer, Heidelberg (2002)
4. Liu, C.: Logical foundations for reasoning about trust in secure digital communication. In: Stumptner, M., Corbett, D.R., Brooks, M. (eds.) *Canadian AI 2001. LNCS*, vol. 2256, pp. 333–344. Springer, Heidelberg (2001)
5. Gong, L., Needham, R., Yahalom, R.: Reasoning about belief in cryptographic protocols. In: Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy, pp. 234–248. IEEE Computer Society Press, Los Alamitos (1990)
6. Oorschot, P.C.V.: Extending cryptographic logics of belief to key agreement protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 233–243. ACM Press, New York (1993)
7. Syverson, P.F., Oorschot, P.C.V.: A unified cryptographic protocol logic. *NRL Publication* (1996)
8. Abadi, M., Tuttle, M.R.: A semantics for a logic of authentication (extended abstract). In: *PODC*, pp. 201–216 (1991)
9. Moser, L.: A logic of knowledge and belief for reasoning about computer security. In: Proceedings of the Computer Security Foundations Workshop II, pp. 57–63. IEEE Computer Society Press, Los Alamitos (1989)
10. Wedel, G., Kessler, V.: Formal semantics for authentication logics. In: Martella, G., Kurth, H., Montolivo, E., Bertino, E. (eds.) *ESORICS 1996. LNCS*, vol. 1146, pp. 219–241. Springer, Heidelberg (1996)
11. Yahalom, R., Klein, B., Beth, T.: Trust relationships in secure systems - a distributed authentication perspective. In: Proceedings of the 1993 IEEE Symposium on Research in Security and Privacy, pp. 150–164 (1993)
12. Boyd, C., Mao, W.: Designing secure key exchange protocols. In: Gollmann, D. (ed.) *ESORICS 1994. LNCS*, vol. 875, pp. 93–105. Springer, Heidelberg (1994)
13. Bacchus, F., Grove, A.J., Halpern, J.Y., Koller, D.: From statistical knowledge bases to degrees of belief. *CoRR cs.AI/0307056* (2003)
14. Gabbay, D., Pigozzi, G., Woods, J.: Controlled revision – an algorithmic approach for belief revision. *Journal of Logic and Computation* 12(1), 3–22 (2003)
15. Weydert, E.: Ranking revision reloaded. In: *Formal Models of Belief Change in Rational Agents* (2007)
16. Smets, P., Ristic, B.: Kalman filter and joint tracking and classification based on belief functions in the tbm framework. *Information Fusion* 8(1), 16–27 (2007)
17. Ma, J., Orgun, M.A.: Formalising theories of trust for authentication protocols. *Information Systems Frontiers* 10(1), 19–32 (2008)

18. Finger, M., Gabbay, D.M.: Adding a temporal dimension to a logic system. *Journal of Logic, Language and Information* 1, 203–233 (1992)
19. Liu, C., Ozols, M.A., Orgun, M.A.: A fibred belief logic for multiagent systems. In: Zhang, S., Jarvis, R. (eds.) *AI 2005*. LNCS, vol. 3809, pp. 29–38. Springer, Heidelberg (2005)
20. Orgun, M.A., Governatori, G., Liu, C.: Modal tableaux for verifying security protocols. In: *Proceedings of the Formal Approaches to Multi-Agent Systems, FAMAS 2006*, pp. 31–46. Riva del Garda, Italy (2006)
21. Orgun, M.A., Liu, C., Governatori, G.: Modal tableaux for verifying stream authentication protocols. *Autonomous Agents and Multi-Agent Systems* 19(1), 53–75 (2009)
22. Liu, C., Orgun, M.A.: Dealing with multiple granularity of time in temporal logic programming. *Journal of Symbolic Computation* 22(5/6), 699–720 (1996)
23. Gabbay, D.M.: Fibring logics. *Journal of Logic, Language and Information* 9(4), 511–513 (2000)
24. Kracht, M., Wolter, F.: Properties of independently axiomatizable bimodal logics. *The Journal of Symbolic Logic* 56(4), 1469–1485 (1991)
25. Gabbay, D.M., Shehtman, V.B.: Products of modal logics, part 1. *Logic Journal of the IGPL* 6(1), 73–146 (1998)
26. Liu, C., Ozols, M.A., Orgun, M.A.: A temporalised belief logic for specifying the dynamics of trust for multi-agent systems. In: Maher, M.J. (ed.) *ASIAN 2004*. LNCS, vol. 3321, pp. 142–156. Springer, Heidelberg (2004)
27. Liu, C., McLean, P., Ozols, M.A.: Combining logics for modelling security policies. In: *ACSC*, pp. 323–332 (2005)
28. Gabbay, D.M.: *Fibring Logics*. Oxford Logic Guides, vol. 38. Oxford University Press, Oxford (1998)
29. Orgun, M.A., Ma, J., Liu, C., Governatori, G.: Analysing stream authentication protocols in autonomous agent-based systems. In: *Proceedings of the 2nd International Symposium on Dependable Autonomic and Secure Computing*, pp. 325–332. IEEE Computer Society Press, Los Alamitos (2006)
30. Kripke, S.A.: Semantical considerations on modal logic. *Acta Philosophica Fennica* 16, 83–94 (1963)
31. Ma, J., Orgun, M.: Tableaux-based proof techniques for verifying multi-agent systems. In: *Proceedings of the 2008 IEEE International Conference on Distributed Human-Machine Systems (DHMS)*. IEEE Computer Society Press, Los Alamitos (2008)
32. SPORE: Security protocols open repository (2009), <http://www.lsv.ens-cachan.fr/Software/spore/> (last accessed April 14)
33. Anderson, R.J., Needham, R.M.: Programming satan’s computer. In: *Computer Science Today*, pp. 426–440 (1995)
34. Lowe, G.: A family of attacks upon authentication protocols. Technical Report 1997/5, Department of Mathematics and Computer Science, University of Leicester (1997)
35. Steiner, J.G., Neuman, B.C., Schiller, J.: Kerberos: An authentication service for open network systems. In: *Proceedings of the Winter 1988 Usenix Conference*, pp. 191–202 (1988)

Trust and Reputation Policy-Based Mechanisms for Self-protection in Autonomic Communications

Martin Serrano¹, Sven van der Meer¹, John Strassner¹, Stefano De Paoli²,
Aphra Kerr², and Cristiano Storni³

¹ Waterford Institute of Technology, Telecommunications Software and Systems Group,
ArcLabs Ireland, West Campus, Waterford Co., Ireland
{jmserrano, vdmeer, jstrassner}@tssg.org

² National University of Ireland, Maynooth, Sociology Department
Maynooth Campus, Maynooth, Ireland
{stefano.depaoli, aphra.kerr}@nuim.ie

³ University of Limerick, Department of Computer Science and Information Systems
Interaction Design Centre, Limerick, Ireland
cristiano.storni@ul.ie

Abstract. Currently, there is an increasing tendency to migrate the management of communications and information systems onto the Web. This is making many traditional service support models obsolete. In addition, current security mechanisms are not sufficiently robust to protect each management system and/or subsystem from web-based intrusions, malware, and hacking attacks. This paper presents research challenges in autonomic management to provide self-protection mechanisms and tools by using trust and reputation concepts based on policy-based management to decentralize management decisions. This work also uses user-based reputation mechanisms to help enforce trust management in pervasive and communications services. The scope of this research is founded in social models, where the application of trust and reputation applied in communication systems helps detect potential users as well as hackers attempting to corrupt management operations and services. These so-called “cheating services” act as “attacks”, altering the performance and the security in communication systems by consumption of computing or network resources unnecessarily.

Keywords: Trust Management, Pervasive Services, Policy-Based Management, Autonomic Communications, Pervasive Computing, Reputation Mechanisms, Systems Management, Social Networks, Information Systems.

1 Introduction

Social relationships are built based on the trust between people. Computing and communications systems are now aiming to take advantage of such models and then use the concepts of reputation and trust to, for example, generate systems offering trustworthy and secure information services and networking applications. Such systems, as trust generators, can also be used to support diverse applications in other

systems or sub-systems requiring certain security levels. In computing, trust management arise from the necessity to remotely execute operations, and has been adopted as a way to enable security for distributed systems in situations where risk taking management decisions exists. Hence, trust management systems must offer certain guarantees to securing information, as well as processes that create, manage distribute, and govern information and services, in a reliable and efficient manner.

Trust management [1] is based on a philosophy of decentralizing security decisions, and as consequence of this, the creation of open and decentralized systems and stable and secure services [2] are promoted. In current service management systems and Future Internet solutions is crucial to protect the system and its sub-systems. Actually, there are several initiatives focused on specifying how to build open, distributed and secure management systems. The NGOSS, or New Generation Operations Systems and Software from the Tele-Management Forum (TMF) [3], attempts to standardize the processes and data used by Business and Operations Support Systems (BSSs and OSSs) for example. However, even ambitious initiatives such as this have failed to produce information models that are able to provide trust management and reputation services. Without a standard definition of such concepts, vendors will build their own device- and application-specific data models that will redefine common concepts. As information and management of communications systems migrating onto the web, the adoption of trust management practices is crucial to protect information and processes that have an inherent risk associated with it.

The use of a services-oriented philosophy helps this problem, and enables service support models to evolve and meet the needs of new applications that incorporate new technologies. The DEN-ng information model [4][5][6] was built using many different abstractions following this philosophy, and forms the basis for the work presented in this paper. However, the development of a robust information-centric view is only one part of the solution. The evolution of current security mechanisms in systems is not sufficiently effective to protect each management system and/or subsystem from intrusions or hacking attacks, especially if web-based operation is desired. This requires dedicated trust management protocols, formats, applications, and tools. In addition, trust management plays, more than ever, an important role in the design of any system and the interfaces with the user(s).

Communications and Internet systems have not yet sufficiently addressed the importance of the *social* needs that users and the adaptation of their social models have in computing systems, however some initiatives following this translation between domains exist. A clear example of such translation of models can be found in bio-inspired systems, where biological reactions from the human body or animals are studied and implemented as computing mechanisms emulating such behavior [7]. Another example is the operation of social networks, where features and human behaviors are implemented in communications networks and systems. So, the awareness of such social models and the necessity of using them to provide an immersive environment generating trust in computing systems is required.

The capture of such models and its adequate translation and implementation into computing environments has acquired more attention in the trust management community. Trust management is broadly accepted as required for modeling, analyzing, and managing decisions within certain trust levels [8][9]. This paper presents an approach, rooted in the management of pervasive systems, where autonomic

management is shown to be a promising approach to implementing self-protection. Figure 1 depicts our vision about typical web service security in the left-hand side of the picture supporting management communications. Our approach specifically examines how to support services and network security in pervasive services and the Future Internet. This trust management approach concentrates on management services and applications using an autonomic orientation.

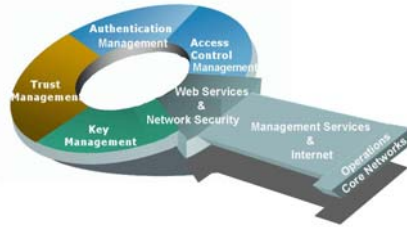


Fig. 1. Management of Services to Enable Trust in Future Internet using Autonomic Technologies

This paper discusses a methodology that can be used in the framework of trust management to create solutions using reputation mechanisms based on policies. This approach can then apply this knowledge to support dynamic management of pervasive services. The reputation mechanism proposed follows social networks and other user-based reputation management systems principles [10][11]. The shortcomings of such systems, in terms of multi-criteria analysis and evaluation as well as implementation and realization experiences, are addressed in this paper, with the objective to illustrate how this research activity can develop new solutions that satisfy the important real world requirements of using multi-criteria for computing appropriate levels of reputation and trust using policy-based management mechanisms.

The rest of the paper is organized as follows. Section 2 describes related work for offering efficient and secure service deployment using trust management operations. Section 3 briefly describes the interaction between users and systems, and then introduces trust management based on reputation models that describe its conceptual relationship with the policy-based management paradigm. Section 4 introduces our policy-based trust model in services support as well as in pervasive management operations. Section 5 introduces a scenario in which policies and the trust and reputation model proposed are used for validation purposes. Finally, section 6 summarizes the contributions and conclusions of this paper.

2 Related Work

Participative user design [12] has been strongly influenced by ubiquitous computing, which is in turn motivated by developing systems with the ability to incorporate surrounding information about users and the environment, and to use such information to perform operations as described in [13][14][15]. However, these efforts do not usually include the use of trust management concepts.

There are a significant number of approaches that use social models for defining secure interactions between users and computational systems [16][17]. Recently, in the field of management services, user behavior has been described and introduced for taking control of specific management operations in networks and the systems [18]. Other approaches explore the translation of human behavior using social models, which enables systems to control services in a more secure and transparent manner [19]. An example is the NetTrust project [20] that uses a value-sensitive design mechanism to validate trust levels, particularly for e-commerce applications.

This approach can be applied to many other systems, such as [21] and [22]. Our research extends these approaches and concentrates on the task of supporting trust and reliable management service operations. Trust management is crucial in the deployment of pervasive secure services and their dynamic management nature allows delegating decision-making. Trust management helps to generate reliable management systems supported by self-protection and autonomic mechanisms.

Approaches for managing trust can be categorized in two major fields, as classified in [23]. However, when related to the field of autonomic communications, both major fields are in some way complementary each other. Today, with most of the services tendency towards a service-based design, the development of trust management follows a more integrated perspective for managing trust, and focuses on providing security and reliability about and for an entity. The use of policies to address this challenge in trust management is relevant; however, policies traditionally manage the decisions of a system for controlling specific set of operations that are pre-defined or pre-programmed. Policies can assist in making decisions when a certain level of ambiguity in the decision-making mechanism is present by utilizing the results of trust management systems.

Policies, as a tool for managing networks and services, have promoted a number of approaches for controlling such operations [24]. The main policy models used in network management are: 1) the IETF policy model [25][26], 2) the DMTF CIM [27], 3) the TMF SID [28], and 4) the DEN-ng in ACF [6]. We use the DEN-ng model because of the reasons documented in [6]. Conceptually, the semantics of a DEN-ng policy rule are: WHEN a set of events *triggers* the evaluation of a set of conditions, IF those conditions evaluate to TRUE, THEN execute a set of actions. Optionally, a set of alternative actions can be executed if the evaluation of the condition is FALSE. Our research uses policies as the mechanism to produce dynamic control and changes in management, orchestration of services, and performance of systems [5][6][29].

3 Trust and Reputation Model

Our current research is based on extending our policy representation to enable it to be used in trust management scenarios. This novel research task relates two different application fields – context awareness and trust management - by using contextual information from social models to determine reputation and trust values that can then guide the deployment of service offerings.

In pseudo-code, these mapping relationships are as follow:

The Social Model ... (1)

WHEN an *User* is requesting a service,
IF an *Evaluator* evaluates *User* can be **TRUSTED**,
THEN allows to *Execute* activities,
ELSE apply *Restrictive* actions.

The Trust Policy-Based Model ... (2)

WHEN an *event_clause* from a *User* is received,
 (which is able to trigger a *condition_clause* evaluation)
IF a *condition_clause* evaluates to **TRUE**,
 (subject to the evaluation strategy)
THEN execute one or more *actions*,
 (subject to the rule execution strategy)
ELSE execute alternative one or more *actions*,
 (subject to the rule execution strategy)

An *event_clause* specifies the event or set of events that trigger the evaluation of the *condition_clause* of the policy rule. A *condition_clause* evaluates the condition or set of conditions in order to determine which, if any, of the set of actions should be executed in response to the triggering event(s). An *action_clause* specifies the set of actions to be executed if the result of the *condition_clause* evaluates to **TRUE** (optionally, a second *action_clause* can be defined to specify the set of actions to be executed if the *condition_clause* evaluates to **FALSE**). In our policy model, the concept of restrictions arising from a lack of trust is also represented as actions.

Previous work has modeled an enhanced version of role-based access control using the DEN-ng policy model [30]. This work enables us to extend the DEN-ng policy model to include trust management concepts. Another important feature of our previous work is the concept of the Policy Continuum [4][31]. This is an abstraction that enables different concepts and terminology to be used to define policies for different constituencies (e.g., business users, architects, and programmers), and relates these policies through a set of transformations. This is an important tool to represent multi-criteria decisions and different levels of trust, in which different criteria and trust concepts for different constituencies can be related to each other.

The users of future communication systems should be able to interact with systems more freely, and should be able to configure their own services according to personal preferences and needs. This has the unfortunate side effect of encouraging a larger number of malicious users to try to cheat or possibly disable the system. For these and other reasons, it is important to create mechanisms that help detect such attacks and differentiate between trusted and malicious users. To do so, we propose to use policies that incorporate trust and reputation mechanisms.

Reputation mechanisms can be considered to be a subset of trust management systems that assign a computable measure of trust to any entity on the basis of the past history of that entity. The existing approaches in reputation-based trust management systems are not very different from that of social scientists. For example, [32] observed that participants in a trust relationship thought of trust as follows: "We wish to know the sort of person we are dealing with before we deal with him. But we will

know it only imperfectly”. For this reason, [33] concluded: “Prior to the Internet, such questions were answered, in part, through personal and corporate reputations.

Vendors provided references, Better Business Bureaus tallied complaints, and past personal experience and person-to-person gossip told you on whom you could rely and on whom you could not.” The point made by [34] is that questions and concerns related to the trustworthiness and reliability of users and other entities over the Internet seems to be more challenging than in off-line relationships. The issue is therefore how to know the past behavior of an entity, how to compute and incorporate that behavior into calculations to define current trust levels, and then decide which rules to use to determine whether to trust or not such an entity.

The trust model is based on certain levels of reputation and, as it happens in real life, people trust in other people. However, there are questions that arise, including (1) what kind of reputation levels are necessary to evaluate the trustworthiness of a user, (2) what criteria must be considered when a trust value of the reputation of the users is assigned, and (3) what happens if trust depends on a combination of several criteria.

The mechanisms based on reputation and trust, for example, have been broadly used in many and diverse on-line sales and auction models, such as eBay, Amazon and Expansys. In this context, when a buyer wants to purchase a product, the buyer must make a decision based on the description and reputation of the seller. Most of the time, this is defined as a percentage, being the trust level average assigned by other buyers about the sellers. However, what happens if the seller is new to the market and does not have an established reputation? How can a buyer determine if sellers are cheating buyers by evaluating themselves positively? Hence, the final decision in trust is delegated to the buyers, mainly because there are not efficient mechanisms implemented that, based on trust or reputation can help the users and/or the systems to make such decisions. We use policies to control service operations and activities based on statistical variations that indicate untrusted operations in the system and then apply actions as restrictions. We evaluate the individual reputation values and the combined values (reputation values with statistical variations) to provide a more accurate trust value; this is used by policy-based trust management mechanisms to offer more reliable decisions to the systems based on the revised reputation values.

Multi-criteria is an important issue in trust management. We address this issue by using policy-based management. Figure 2 represents multi-criteria being used to

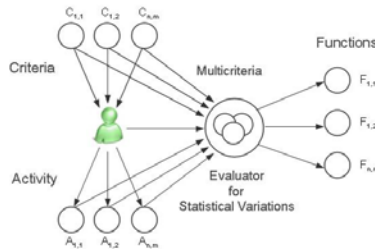


Fig. 2. Policy-Based Approach for Solving Multi-Criteria Problems

define the trust level being assigned to users. An evaluator uses reputation and statistical variations to evaluate and control if the user is trusted or not. If the user is determined to be trusted, then that user is allowed to execute authorized function(s).

The functional component acting as an evaluator in Figure 2 has the capability to analyze and provide actions as a result of this analysis. For example, when management applications must perform network changes, policy rules can take into account static as well as dynamic end user criteria and activities. Examples of criteria are user identity and electronic keys, while examples of activity are the statistical results such as visiting specific sites (e.g., eBay or Amazon) or contracts with services (e.g., phone and broadband services). Thus, users will be able to modify and execute system's actions according reputation performance, which is enforced using policies.

3.1 Premises in Trust Management

Typical approaches for security in the systems are based on passwords or key codes, and these techniques work well on closed systems. However, when decentralized security systems are used, public key infrastructure (PKI) mechanisms [35] emerge as more suitable solutions that provide decentralized and more secure models. Cryptography and digital certificates [36] are now used in many security solutions. Digital certificates ensure with an increased level of security, the transfer of information. The challenge is to decide who can access what type of information.

Sociological research [37] views trust as a relationship between individuals (for example between persons) or collective (for example Nation States) social actors. Trust relationships enable social actors to take decisions that have some amount of risk, in a situation in which there is a lack of knowledge and the possibility to make an informed choice is precluded. This means that if we look at the pseudo code describing the social model in Section 2 above, in real social situations, the roles of *User* and *Evaluator* are filled by appropriate social actors (i.e., by persons). In sociological literature, these roles are called the *Trustor* (the entity who places trust, the *Evaluator*) and the *Trustee* (the actor which receives trust, the *User*). When trust is placed in the *Trustee*, the *Trustor* is able to solve uncertain situations by choosing one of several alternatives, each based on trust and reputation. An example is when a user (*Trustor*) wants to use an on-line banking system (*Trustee*). How can the user be sure that the online banking system, which represents a bank, is a reliable banking system? The user cannot know this beforehand, because the user cannot prove that what he or she thinks is a banking system is not instead a forgery. Here is when a trust relationship must be established.

In this case, the recommendation of other people, based on previous experiences, and/or other evidence, can be used to establish a trust relationship. This can then be tested by, for example, conducting a small banking transaction and using the online system and then verifying the correctness of that transaction by physically visiting the bank. In other words, a trust relationship involves risk as well as uncertainty.

The same model applied to Trust Management in autonomic computing solutions needs to take in account that the *Evaluator* (or *Trustor*) in a trust relationship is a Trusted System (i.e., a machine making decisions of behalf of human beings and controlling their actions). Therefore, in Trust Management, the decisions of users to

trust systems are delegated to the Trusted System [38]. Along this line of reasoning, it is important to recognize that the enacted evaluation strategy satisfying the IF condition is never a neutral one.

Security policies and the mechanisms enforcing them might, for example, determine that there are unnecessary divisions of labor or unwanted social discriminations (e.g., gender, racial or age discriminations) in relation to how information is accessed. For these and other reasons, systems must be assisted in making decisions for determining the trustworthiness of users as well as in detecting cheating users. In addition, these systems must also be evaluated in their actions. The systems must take appropriate actions, such as restricting operations and/or blocking some or all user activities that could be performed. The service management tasks can then offer more dynamic performance and more efficient operation.

3.2 Methodology and Formal Approach

In this section, we describe the general concepts of the trust and reputation mechanism that uses policies to evaluate and define user capabilities to create, use and deploy web-based and Internet services. The reason to use policies is founded in the benefits that policies provide when they are used to control pervasive services [4][39]. We implement secure policies using an autonomic solution approach.

We start from the premise that a user can create, configure and personalize services according to personal requirements and/or needs. For example, Joe wants a broadband service for downloading video on demand on weekends, but a simpler and more cost-effective data service for checking email on weekdays. Key identifiers are created when the service is deployed and are associated (one to him and one to the service). The lowest rate of reputation is assigned to Joe, since he is the creator of the service. However Joe can be a user of other services; this is an activity that can be monitored and studied by systems to adjust Joe's reputation level. Thus, Joe's reputation increases when he uses other services in a reliable manner. This is an automatic way to adjust the reputation value of Joe. However, it does not adjust the trust level of Joe's service – this requires other people to be able to use that service reliably. In fact, it may be that to compute a trust level for Joe's service, multiple criteria must be considered, studied and evaluated.

To increase the trust level, multiple criteria must be considered, studied and evaluated. For example, if Joe provides personal information, or if Joe uses a key identifier already assigned from a trusted source, then those criteria can be used to provide statistics for defining a trust level for Joe. Hence:

$$\text{Criteria } C_{n,m} + \text{Activity } A_{n,m} \rightarrow \text{Functions } F_{n,m} \quad \dots(3)$$

A formal way to capture and study the criteria and activity is to relate them in a matrix, with elements as rows and columns containing first order logic values to categorize and classify the user(s). In other words, if the element in the matrix exists, the value assigned will be 1; if the element in the matrix is not accessible, then the value assigned will be 0. In this way, the matrix is composed as Figure 3 shows.

		Criteria Columns			
		$\bar{a}_{i,j}$	Name	Surname	e-Key ...
Activity Rows	Ebay User	$\bar{a}_{1,1}$	$\bar{a}_{1,2}$	$\bar{a}_{1,3}$...
	Paypal User	$\bar{a}_{2,1}$	$\bar{a}_{2,2}$	$\bar{a}_{2,3}$...
	Amazon User	$\bar{a}_{3,1}$	$\bar{a}_{3,2}$	$\bar{a}_{3,3}$...

Fig. 3. Trust Matrix for Criteria and Activity Allocation

The user extensibility is represented by a single row matrix. Figure 4 shows this formalism, in which this single row matrix is made up of single elements that operate as a scalar product with the user matrix. This provides a scalable representation of our approach.

	User 1	User 2	User 3	...
User Row	u_1	u_2	u_3	...

Fig. 4. User Matrix in a Single Row

The values of the criteria correspond to the number of users. Figure 5 shows the matrix representation.

$\bar{a}_{i,j}$					
$\bar{a}_{1,1}$	$\bar{a}_{1,2}$	$\bar{a}_{1,3}$...	,	u_1
$\bar{a}_{2,1}$	$\bar{a}_{2,2}$	$\bar{a}_{2,3}$...		u_2
$\bar{a}_{3,1}$	$\bar{a}_{3,2}$	$\bar{a}_{3,3}$...		u_3
...

Fig. 5. User Matrix for Multiple Users

We assume that multiplication of two matrices is well-defined only if the number of columns of the left matrix is the same as the number of rows of the right matrix. The number of criteria must correspond to the number of users; this restriction is strictly followed to get the identity matrix. Thus, when the diagonal contains the value “1” in all of its elements, a user is evaluated as a trusted user to operate services and applications.

Figure 6 shows the trust function as a matrix representation for multiple user criteria. In other words, the matrix representation helps to identify possible cases when it is necessary to evaluate multiple-criteria cases. Furthermore, this evaluation is made in combination with the user’s activity represented as statistical values.

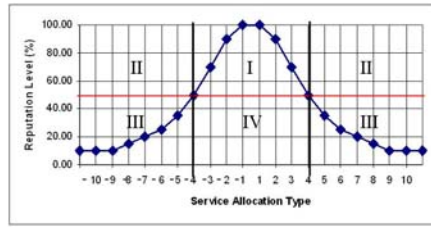


Fig. 8. Regions Related to Policy Service Allocation and User's Reputation

Region I is the most secure region, where the users are trusted all the time. Region IV has users that lack high reputation values, but these users can be considered reliable as they do not attempt to create services and their reputation value is medium. Region II is where the users have created many services, but do not themselves use those services; consequently, these users are trusted but do not have high reputation levels. Region III is where hackers and other malicious users are located; these users have many services they created and therefore, the reputation of each user is low.

The methodology used to create the trust and reputation policy-based model, which can be applied to trust the management of service applications, is to define the trust model representation and the concepts involved according to standard sociological models. Sociological criteria are used to define relationships between the information in the service model described as policies, and the information contained in service management policies.

The formal representation of social models are expressed in policy-based form and integrated as classes into the object-oriented policy-based management system. The policy model function defining the number of pervasive management operations in reference with the number of policies is shown in (4), details are described in [5], we associate the management operations with the activity of the user, thus in this way are generated the values used for statistic operations.

$$\sum_{X_s=1}^{ps+pn} F[\{(Ct_n)_m\}\{(Xs_n)_m\}]I^{(ps+pn)} \rightarrow \text{Service Operations} \quad \dots(4)$$

where ps = number for initial service policies,

pn = number of total service policies, and

Xs = service function; Ct = content function for values of $n \geq 1$ and $m \geq 1$

The advantage of combining policy-based management and reputation approaches is the possibility to define values based on the statistical variations in the reputation level and use those values in policies that define the trust level of the user. Figure 9 depicts an example where the number of sub-regions is more specific as a consequence of many users generating services. The policies here assist in evaluating regions and sub-regions according to the same criteria already defined for each of the four main regions above. The number of policies follows the function used in (4) to calculate service operations and define the user activity according to the set of services that the user is using.

The probability is calculated based on specific sub-regions of the four main regions already defined, according to the policy model function in [5], and includes the policy model to construct the trust model for services support.

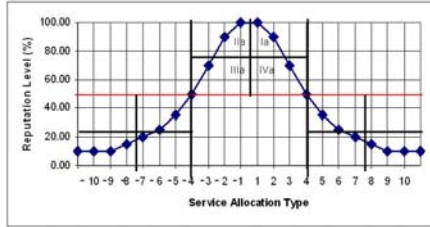


Fig. 9. Sub-regions Related to Policy Service Allocation Number

The policies also assist in supporting decisions based on multiple criteria. Put another way, policies are used to evaluate a diversity of opinions from different users about the same service. The multiple criteria evaluation is done according to personal experiences when using the service provides a level of reputation.

4 Policy-Based Trust Management Model

We support the idea that using social trust models for systems in which users communicate and make decisions having an inherent risk will enhance the security of such systems. A social model of trust needs to be used in order to provide a robust information model that is able to represent user information in a formal manner. Once this is done, this model can then be used in autonomic systems.

Figure 10 illustrates the challenge and the scope of applying trust management multi criteria results. From sociology, we have a social model of trust based on reputation, whereas from technology, we have the ability to create new services. When these two studies are combined a more powerful services composition to trustworthy users can be assigned. The definition of the trust model based on reputation to be used by policy-based systems can make use of the information to generate new services. Therefore, applying such a trust model requires different assignments of metadata to describe trusted users as well as malicious users.

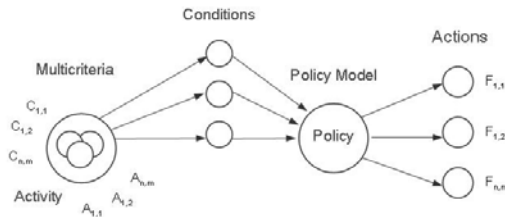


Fig. 10. Policy-Based Model and Trust Concepts Relationships

4.1 Policy-Based Descriptions

The user's information should be used to generate personalized services that reside in the service management system but are enabled by the service operators. Both service creators as well as operators should be able to determine if the services are performing properly according to both the service model definition as well as with the proposed trust model. In this study the use of the DEN-ng policy information model [29] facilitates the inclusion of business goal policies, and utility functions. The DEN-ng *PolicyRule* class represents an intelligent container that gathers metadata and at least one (or more) *PolicyEvent*, *PolicyCondition*, and *PolicyAction*.

Figure 11 shows the definition of Management Policies used in this approach; note that (1) Management Policies may use *any* type of structural representation of a Policy Rule, since the *ManagementPolicy* is an intelligent container that aggregates *PolicyRuleStructures*; (2) the concepts of *PolicySubject* (a set of *ManagedEntities* that requests and/or invokes policies in a holistic manner from and/or on a *PolicyTarget*) and *PolicyTarget* (a set of *ManagedEntities* that a set of policies will be applied to).

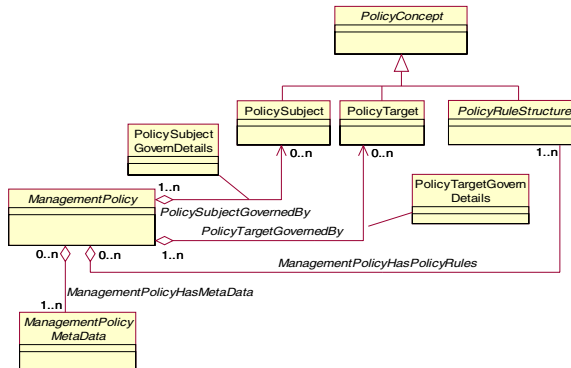


Fig. 11. Simplified DEN-ng ManagementPolicy Model

In DEN-ng, a *ManagedEntity* is something of interest that can be managed. Any *ManagedEntity* can have contextual information associated with it. We model context [40] as an overall concept (the Context class) that may be made up of a set of independently manageable aspects. The associations relating *ManagedEntity* to Context and *ContextData* are both optional. *ContextDataFacts* and *ContextDataInferences* are intelligent containers that house facts and inferences, respectively, that are computed by external applications. Both the Context and *ContextData* classes use the composite pattern for flexibility and extensibility. The *ContextAtomic* and *ContextDataAtomic* classes represent context that can be modeled as a single, stand-alone object. In contrast, the *ContextComposite* and *ContextDataComposite* classes represent context objects that are made up of multiple distinct Context or *ContextData* objects that can each be separately managed. Hierarchies of context information can be defined and related to other context information through the *HasContextData* aggregation. For example, the Context object “Communication” could have the following *ContextData* objects associated with it: PSTN, *CellularDevice*, PDA, and *ComputerDevice*, to model the characteristics of

fixed telephone lines, mobile phones, PDAs, and computers, respectively. Each of these four classes of device uses different types of media and provides different types of communication experiences, and hence different contexts.

The purpose of the *ContextDataDetails* association class is to define the particular semantics of how *ContextData* relates to Context. This enables different types of *ContextData*, each modeling a specific aspect of an overall Context, to be aggregated together with their own semantics. The *ContextSemantics* class represents data and/or knowledge that describes the behavioral aspects of the Context that this *ManagedEntity* is associated with. A similar class (*ContextDataSemantics*) is constructed for the *ContextData* hierarchy. These two classes represent a convenient point for fusing information from ontologies with data from information and data models. For example, machine-based reasoning can now be used with both of these data. They also present convenient points for either augmenting context information (e.g., tagging it with metadata to enhance information retrieval) and/or using context data to perform (for example) a set of services. Finally, these two semantics classes enable the application to declare what it needs to complete its view of context, as opposed to merely obtaining context information.

Figure 12 shows a simplified view of the DEN-ng context-aware policy model. Its purpose is to relate context changes to policy changes by selecting the set of Policy Rules that are appropriate for this particular context. Those Policy Rules are then applied by the autonomic manager to govern system behaviour. The *SelectsPolicies* aggregation defines a given set of Policies that should be loaded based on the current context. Hence, as context changes, policy can change accordingly, enabling our system to adapt to changing demands. The *PolicyResultAffectsContext* association enables policy results to influence Context.

The selected working set of Policies uses the *GovernsManagedEntityRoles* aggregation to define the appropriate roles of the *ManagedEntities* that are influenced by this Context; each *ManagedEntityRole* defines functionality of the *ManagedEntity* that can take on that role.

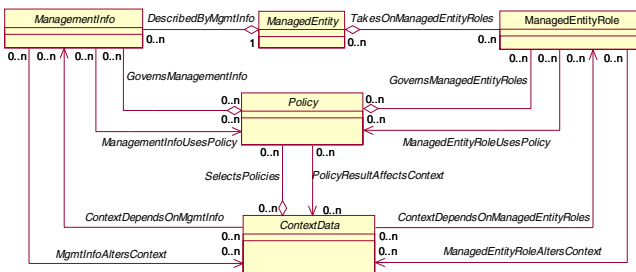


Fig. 12. Simplified DEN-ng Context-Aware Policy Model

Specifically, *Policy* is used to define which management information will be collected and examined; this management information affects policy decisions, as well as selecting which policies should be used at any given time. Once the management information is defined, then the two associations *MgmtInfo Alters Context* and

ContextDependsOnMgmtInfo codify these dependencies (e.g., context defines the management information to monitor, and the values of these management data affect context, respectively).

4.2 Service Logic Descriptions

The logic defining logic interactions and technological implications between organizations in a pervasive service is a result of using service management operations in form of policies (i.e. Service Code and Policies Distribution, Code Maintenance, Service Invocation, Code Execution and Service Assurance). The service management policies are referenced in this paper and detailed examples can be found in [5]. The service logic description corresponds to the most common management operations; however, a more extended set of functions that better reflect application requirements can be used instead.

A more detailed description of these functions can be found in [5]. Implementing decisions using these logic descriptions constitute a first task of this approach towards creating a tool to support trust management. The use of service management logic when DEN-ng is being used as the policy information model includes descriptions which do not assume 'static' information for expressing user requirements. In contrast, the service management systems can process logic descriptions that can be defined dynamically as a result of user interaction.

Additionally the formal language used for expressing web-services is OWL [41]. One of the advantages of using OWL to describe logic concepts is the availability of formal tools that use OWL for parsing, reasoning and editing. We use OWL to describe information that cannot be expressed with graphic notation. We use OWL as the formal language that describes the graphic representations and logic sentences as part in the policy model supporting trust management.

5 Trust Model Scenario

Trust management is able to provide recommendations to governance systems when choices have an inherent security risk. Our implementation uses OWL as a formal language to represent reputation and trust using first order logic; this enables assertions about reputation and trust to be *proven*. This approach also uses policy-based management as the mechanism to execute and enforce decisions in pervasive service operations. Figure 13 shows the scope of test scenarios. These scenarios are focusing on controlling pervasive services and Internet services. System management is supported by a trust generator subsystem with the objective to decentralize access decisions, in this way the management systems delegates security decisions with the objective for improving the management operations.

The most important objectives of our trust- policy-based management system are: (1) to decentralize decisions necessary to ensure proper operation, and (2) to support the deployment of new services in policy-based management systems that can use user information from users who are trustworthy, according to the trust model, to generate new services following the preferences and requirements of those users.

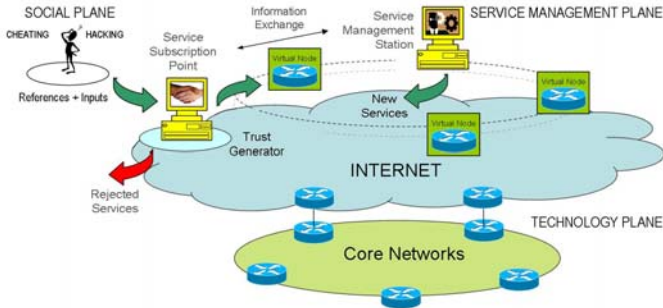


Fig. 13. Mapping of Trust Model into Web-Services and Internet

An important type of service that we aim to support with multicriteria support is what sociologists have called *swift trust* [38]. This is a type of trust deployed by a social actor in temporary systems such as online communities or temporary working teams. In our scenarios, we use this model to support temporary services that, for example, only last for a particular session. This reflects the idea that people depend on this trust model as the alternative to avoid an indeterminate amount of time while measurements are collected that enable the system to compute who can be trusted.

In addition, communications systems must offer trust management and security in the operations performed and governed by management systems. However, this is a challenging trade-off between security and performance, due mainly to the use of proprietary information and data models to design and guide the implementation of securing the information and its associated management processes. Since there is no one common information or data modeling standard that can represent vendor-specific management and security data, securing systems remains a stovepipe design process. This is exacerbated by the increasing number of diverse technologies, each with their own associated operational and management data. Just as there will never be one programming language that all applications will use, there will never be a single information model that all vendors and application developers will use.

This scenario is based on trust management concepts, and uses policy-based principles to provide trust management mechanisms based on user reputation. The scope of this research is founded in real world social scenarios, where the application of trust and reputation models acts as inputs in communication and service management systems to detect potential users attempting to create cheating services.

6 Conclusions

This paper describes the research challenges for trust management in an environment that uses policy-based management to build reputation and trust using social models. The main research contribution is our approach to support multi-criteria reputations in trust management with policies-based mechanisms that offer a formal alternative for representing and implementing guide for trust management.

This research work proposes to decentralize management decisions by using a user-based trust and reputation mechanism. The main application of this model is in

the framework of autonomic solutions for future Internet services. However, the scope of the model is not limited to this scenario; it can be extended for any system requiring decentralized decisions with multi-criteria processes as generator of trust.

Social trust and reputation models are applied in communication systems in order to provide guidance for trusted users to make decisions having a security risk, as well as to help the system detect malicious users and hackers attempting to create cheating services or other disruptive services. Future research is being conducted for developing and comparing different implementations of the model and the statistical results are applied on simulations for decentralized management tasks.

Acknowledgements

This research activity is being funded by High Education Authority (HEA) into the PRTL Cycle 4 research program in the framework of the project *Serving Society: Management of Future Communications Networks and Services*.

References

- [1] Ruohomaa, S., Kutvonen, L.: Trust Management Survey. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) *iTrust 2005*. LNCS, vol. 3477, pp. 77–92. Springer, Heidelberg (2005)
- [2] Khare, R., Rifkin, A.: Trust management on the World Wide Web. *Computer Networks and ISDN Systems Archive* 30, 651–653 (1998)
- [3] The NGOSS Technology Neutral Architecture, TMF 053, Version 5.7 (November 2006)
- [4] Strassner, J.: *Policy Based Network Management*. Morgan Kaufmann, San Francisco (2004)
- [5] Serrano, J.M., Serrat, J., Strassner, J., Foghlú, M.Ó.: Facilitating Autonomic Management for Service Provisioning using Ontology-Based Functions & Semantic Control. In: 3rd IEEE International Workshop on Broadband Convergence Networks (BCN) 2008 in IEEE/IFIP NOMS 2008, Salvador de Bahia, Brazil, April 07-11 (2008)
- [6] Strassner, J.: Introduction to DEN-ng., Tutorial for FP7 PanLab II Project (January 21, 2009)
- [7] Dressler, F., Carreras, I.: *Advances in Biologically Inspired Information Systems: Models, Methods, and Tools*. Springer, Heidelberg (2007)
- [8] Blaze, M., Feigenbaum, J., Lacy, J.: Decentralized Trust Management. In: *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, Los Alamitos, California, USA, pp. 164–173. IEEE Computer Society Press, Los Alamitos (1996)
- [9] Blaze, M., Feigenbaum, J., Resnick, P., Strauss, M.: *Managing Trust in an Information-Labeling System*. European Transactions on Telecommunications (1997)
- [10] Camp, J., Genkina, A., Friedman, A.: *Social and Network Trust*, DIMACS, April 14-15, 2005. DIMACS Center, CoRE Building, Rutgers University, Piscataway, NJ (2005)
- [11] Camp, J.: *Trust and Risk in Internet Commerce*, p. 293. MIT Press, Cambridge (2000)
- [12] Mumford, E.: Participative Systems Design: Practice and Theory. *Journal of Occupational Behaviour* 4(1), 47–57 (1983)
- [13] Abowd, G.D., Dey, A.K., Orr, R., Brotherton, J.: Context-awareness in wearable and ubiquitous computing. In: *Intl. Symposium on Wearable Computers*, pp. 179–180 (1997)
- [14] Brown, P.J., Bovey, J.D., Chen, X.: Context-Aware Applications: From the laboratory to the Marketplace. *IEEE Personal Communications*, 58–64 (1997)

- [15] Chen, G., Kotz, D.: A survey of context-aware mobile computing research, Technical Report, TR2000-381, Department of Computer Science, Dartmouth College (November 2000)
- [16] Brabham, D.C.: Crowdsourcing as a Model for Problem Solving. An Introduction and Cases. *Intl. Journal of Research into New Media Technologies* 14(1) (2008)
- [17] MacLean, et al.: User-Tailorable Systems: Pressing the Issues with Buttons. In: *Proceedings of CHI, Conference on Human Factors in Computer Systems* (1990)
- [18] Li, H., Zhang, X., Wu, H., Qu, Y.: Design and Application of Rule Based Access Control Policies. In: *Proceedings of the 10th International Conference on Information and Knowledge Management, Atlanta, GA, USA, November 5-10* (2001)
- [19] Grandison, T., Sloman, M.: Specifying and Analysing Trust for internet Applications. In: *Towards the knowledge Society: eCommerce, eBusiness and eGovernment. The Second IFIP International Conference on E-Commerce, E-Business, E-Government, Lisbon, Portugal* (October 2002)
- [20] NetTrust Project, <http://www.ljean.com/NetTrust/>
- [21] Lamparter, S., Agarwal, S.: Specification of Policies for Automatic Negotiations of Web Services. In: *Proceedings of the 4th International Semantic Web Policy Workshop, Galway, Ireland, November 7* (2005)
- [22] Aberer, K., Despotovic, Z.: Managing Trust in a Peer-2-peer Information System. In: *Proceedings of the 10th International Conference on Information and Knowledge Management, Atlanta, GA, USA, November 5-10* (2001)
- [23] Bonatti, P., Duma, C., Olmedilla, D., Shahmehri, N.: An integration of Reputation-based and policy Trust Management. In: *Proceedings of the 4th International Semantic Web Policy Workshop, Galway, Ireland, November 7* (2005)
- [24] Damianou, N., Bandara, A., Sloman, M., Lupu, E.: A Survey of Policy Specification Approaches, Dept. of Computing, Imperial College of Science Technology and Medicine, London, UK (2002)
- [25] Moore, E., Elleson, J., Strassner, J.: Policy Core Information Model-Version 1 Specification. IETF Request for comments (RFC 3060) (February 2001), <http://www.ietf.org/rfc/rfc3060.txt>
- [26] Moore, E.: Policy Core Information Model-Extensions. IETF Request for comments (RFC 3460) (January 2003), <http://www.ietf.org/rfc/rfc3460.txt>
- [27] DMTF, CIM schema, can be downloaded from http://www.dmtf.org/standards/cim/cim_schema_v220/
- [28] TMF SID schema, members only, can be downloaded from <http://www.tmforum.org/page35501.aspx>
- [29] Strassner, J., Neuman de Souza, J., Raymer, D., Samudrala, S., Davy, S., Barrett, K.: The Design of a New Policy Model to Support Ontology-Driven Reasoning for Autonomic Networking. In: *5th Latino-America Network and Operations Management Symposium (LANOMS), Salvador Bahia, Brazil* (2007)
- [30] Strassner, J., Fu, Z.: Policy Based Enforcement of Ubiquitous Role Based Access Control. In: *4th International IEEE Workshop on Managing Ubiquitous Communications and Services (MUCS), Munich, Germany, May 25* (2007)
- [31] Davy, S., Jennings, B., Strassner, J.: The Policy Continuum – A Formal Model. In: Jennings, B., Serrat, J., Strassner, J. (eds.) *Proc. of the 2nd IEEE International Workshop MACE, Multicon, Berlin. Multicon Lecture Notes, No. 6, pp. 65–78* (2007)
- [32] Dasgupta, P.: Trust as a Commodity. In: *Trust: Making and Breaking Cooperative Relations. Blackwell, Oxford* (1988)

- [33] Resnick, P., Zeckhauser, R., Friedman, E., Kuwabara, K.: Reputation Systems. *Communications of the ACM* 43(12), 45–48 (2000)
- [34] Sztompka, P.: *Trust: A sociological Theory*. Cambridge University Press, Cambridge (1999)
- [35] Schneier, B.: *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd edn. John Wiley and Sons, New York (1996)
- [36] Lampson, B., Rivest, R.: SDSI - A Simple Distributed Security Infrastructure. In: *DI-MACS Workshop on Trust Management in Networks*, South Plainfield, NJ (1996)
- [37] De Paoli, S., Kerr, A.: *Conceptualizing Trust*. NIRSA Working Paper N. 40, National University of Ireland Maynooth (2008)
- [38] Meyerson, D., Weick, K.E., Kramer, R.M.: *Swift Trust and Temporary Group. Trust in Organisations*. Sage, Thousand Oaks (1996)
- [39] Sloman, M.: Policy Driven Management for Distributed Systems. *Journal of Network and Systems Management*, 215–333 (1994)
- [40] Strassner, J., Samudrala, S., Cox, G., Liu, Y., Jiang, M., Zhang, J., van der Meer, S., Foghlú, M.Ó., Donnelly, W.: The Design of a New Context-Aware Policy Model for Autonomic Networking. In: *5th IEEE ICAC*, Chicago, Illinois, June 2-6 (2008)
- [41] De Bruijn, J., Fensel, D., Lara, R., Polleres, A.: *OWL DL vs. OWL Flight: Conceptual Modelling and Reasoning for the Semantic Web* (November 2004)

Author Index

- Brinkschulte, Uwe 47
Burgess, Mark 75
- Cahill, Vinny 105
Camp, L. Jean 1
Chen, Xiangqun 120
Chiarini, Marc 75
Couch, Alva L. 75
- De Paoli, Stefano 249
Dusparic, Ivana 105
- Eilers, Dirk 32
- Guo, Yao 120
- Hähner, Jörg 2
Han, Jun 209
Hitchens, Michael 134
- Jin, Cong 62
- Kerr, Aphra 249
Knorr, Rudi 32
- Li, Jing 209
Li, Lei 221
Long, Xiang 209
- Ma, Dianfu 209
Ma, Ji 236
Müller-Schloer, Christian 2
- Nafz, Florian 17
Nagarajan, Aarthi 134
Niemi, Valtteri 179
- Orgun, Mehmet A. 236
Ortmeier, Frank 17
- Reif, Wolfgang 17
- Sanders, Jeffrey W. 90
Sattar, Abdul 236
Seebach, Hella 17
Serrano, Martin 249
Smith, Graeme 90
Steffen, Marcel 2
Steghöfer, Jan-Philipp 17
Storni, Cristiano 249
Strassner, John 249
Sung, Wen-Tsai 149
- Tomforde, Sven 2
- van der Meer, Sven 249
Varadharajan, Vijay 134, 221
von Renteln, Alexander 47
- Wang, Hua 120
Wang, Yan 221
Weiss, Gereon 32
- Xiang, Yang 164
- Yan, Rong 194
Yan, Zheng 179, 194
Ye, Junmin 62
- Zeller, Marc 32
Zhan, Zemei 62
Zhang, Qingguo 62
Zhu, Wen Tao 164