



# **ICAS 2013**

The Ninth International Conference on Autonomic and Autonomous Systems

ISBN: 978-1-61208-257-8

March 24 - 29, 2013

Lisbon, Portugal

## **ICAS 2013 Editors**

Nikola Serbedzija, Fraunhofer FOKUS, Germany

Petre Dini, Concordia University, Canada / China Space Agency Center, China

# ICAS 2013

## Foreword

The Ninth International Conference on Autonomic and Autonomous Systems [ICAS 2013], held between March 24 - 29, 2013 in Lisbon, Portugal, was a multi-track event covering related topics on theory and practice on systems automation, autonomous systems and autonomic computing.

The main tracks referred to the general concepts of systems automation, and methodologies and techniques for designing, implementing and deploying autonomous systems. The next tracks developed around design and deployment of context-aware networks, services and applications, and the design and management of self-behavioral networks and services. We also considered monitoring, control, and management of autonomous self-aware and context-aware systems and topics dedicated to specific autonomous entities, namely, satellite systems, nomadic code systems, mobile networks, and robots. It has been recognized that modeling (in all forms this activity is known) is the fundamental for autonomous subsystems, as both managed and management entities must communicate and understand each other. Small-scale and large-scale virtualization and model-driven architecture, as well as management challenges in such architectures are considered. Autonomic features and autonomy requires a fundamental theory behind and solid control mechanisms. These topics gave credit to specific advanced practical and theoretical aspects that allow subsystem to expose complex behavior. We aimed to expose specific advancements on theory and tool in supporting advanced autonomous systems. Domain case studies (policy, mobility, survivability, privacy, etc.) and specific technology (wireless, wireline, optical, e-commerce, banking, etc.) case studies were targeted. A special track on mobile environments was indented to cover examples and aspects from mobile systems, networks, codes, and robotics.

Pervasive services and mobile computing are emerging as the next computing paradigm in which infrastructure and services are seamlessly available anywhere, anytime, and in any format. This move to a mobile and pervasive environment raises new opportunities and demands on the underlying systems. In particular, they need to be adaptive, self-adaptive, and context-aware.

Adaptive and self-management context-aware systems are difficult to create, they must be able to understand context information and dynamically change their behavior at runtime according to the context. Context information can include the user location, his preferences, his activities, the environmental conditions and the availability of computing and communication resources. Dynamic reconfiguration of the context-aware systems can generate inconsistencies as well as integrity problems, and combinatorial explosion of possible variants of these systems with a high degree of variability can introduce great complexity.

Traditionally, user interface design is a knowledge-intensive task complying with specific domains, yet being user friendly. Besides operational requirements, design recommendations refer to standards of the application domain or corporate guidelines.

Commonly, there is a set of general user interface guidelines; the challenge is due to a need for cross-team expertise. Required knowledge differs from one application domain to

another, and the core knowledge is subject to constant changes and to individual perception and skills.

Passive approaches allow designers to initiate the search for information in a knowledge-database to make accessible the design information for designers during the design process. Active approaches, e.g., constraints and critics, have been also developed and tested. These mechanisms deliver information (critics) or restrict the design space (constraints) actively, according to the rules and guidelines. Active and passive approaches are usually combined to capture a useful user interface design.

We take here the opportunity to warmly thank all the members of the ICAS 2013 Technical Program Committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to ICAS 2013. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the ICAS 2013 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that ICAS 2013 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the fields of autonomic and autonomous systems.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the charm of Lisbon, Portugal.

**ICAS 2013 Chairs:**

Michael Bauer, The University of Western Ontario - London, Canada

Radu Calinescu, University of York, UK

Michael Grottke, University of Erlangen-Nuremberg, Germany

Bruno Dillenseger, Orange Labs, France

## ICAS 2013

### Committee

#### ICAS Advisory Chairs

Michael Bauer, The University of Western Ontario - London, Canada  
Radu Calinescu, University of York, UK  
Michael Grottke, University of Erlangen-Nuremberg, Germany  
Bruno Dillenseger, Orange Labs, France

#### ICAS 2013 Technical Program Committee

Jemal H. Abawajy, Deakin University, Australia  
Sameh Abdel-Naby, University College Dublin, Ireland  
António Abelha, Universidade do Minho - Braga, Portugal  
Nouara Achour, USTHB University, Algeria  
Carl Adams, University of Portsmouth, UK  
Javier Alonso, Duke University, USA  
Razvan Andonie, Central Washington University - Ellensburg, USA  
Richard Anthony, University of Greenwich, UK  
Eva Ibarrola Armendariz, Escuela Técnica Superior de Ingeniería de Bilbao, Spain  
Ismailcem Budak Arpinar, University of Georgia - Athens, USA  
Tsz-Chiu Au, Ulsan National Institute of Science and Technology (UNIST), Korea  
Roger Azevedo, McGill University, Canada  
Mark J. Balas, University of Wyoming - Laramie, USA  
Michael Bauer, The University of Western Ontario - London, Canada  
Matthias Becker, University Hannover, Germany  
Julita Bermejo-Alonso, Universidad Politécnica de Madrid, Spain  
Daniel Berrar, Tokyo Institute of Technology, Japan  
Philippe Besnard, IRIT - CNRS /Université Paul Sabatier - Toulouse, France  
Ateet Bhalla, Oriental Institute of Science & Technology - Bhopal, India  
Karsten Böhm, Fachhochschule Kufstein, Austria  
Fabienne Boyer, University of Grenoble I, France  
Ruth Breu, University of Innsbruck, Austria  
Daniela Briola, University of Genova, Italy  
David W Bustard, University of Ulster, UK  
Giacomo Cabri, Università di Modena e Reggio Emilia, Italy  
Radu Calinescu, University of York, UK  
Paolo Campegiani, University of Roma Tor Vergata, Italy  
Sara Casolari, Università di Modena e Reggio Emilia, Italy  
Fernando Cerdan, Universidad Politecnica de Cartagena, Spain  
Michal Certicky, Comenius University - Bratislava, Slovakia  
Lei Chen, Sam Houston State University, USA  
Philippe Codognet, University of Tokyo, Japan  
Stéphanie Combettes, SMAC –IRITm Toulouse, France  
Noel de Palma, University Joseph Fourier, France

Marina De Vos, University of Bath, UK  
Sotirios Diamantas, Pusan National University, South Korea  
Tadashi Dohi, Hiroshima University, Japan  
Carlos Duarte, University of Lisbon, Portugal  
Larbi Esmahi, Athabasca University, Canada  
Thaddeus Eze, University of Greenwich - London, UK  
Ziny Flikop, Scientist, USA  
Naoki Fukuta, Shizuoka University, Japan  
Wai-keung Fung, University of Manitoba - Winnipeg, Canada  
Alex Galis, University College London, UK  
Rodrigo Garcia Carmona, Universidad Politecnica de Madrid (DIT-UPM), Spain  
Fabio Gasparetti, Roma Tre University, Italy  
Joseph Giampapa, Carnegie Mellon University, USA  
Andrzej M. Goscinski, Deakin University - Geelong, Australia  
Dominic Greenwood, Whitestein, Switzerland  
William Grosky, University of Michigan - Dearborn, USA  
Michael Grottke, University of Erlangen-Nuremberg, Germany  
Jordi Guitart, Barcelona Supercomputing Center (BSC), Spain  
Takako Hashimoto, Chiba University of Commerce, Japan  
Wladyslaw Homenda, Warsaw University of Technology, Poland  
Marc-Philippe Huget, Polytech Annecy-Chambery-LISTIC | University of Savoie, France  
Tim Hussein, University of Duisburg-Essen, Germany  
Yoshiro Imai, Kagawa University, Japan  
Luis Iribarne, University of Almeria, Spain  
Yiming Ji, University of South Carolina - Beaufort, USA  
Yanguo Jing, London Metropolitan University, UK  
María José Ibáñez, RIAM I+L Lab (GNOSS), Spain  
Hamed Ketabdar, Deutsche Telekom Laboratories / TU Berlin, Germany  
Fernando Koch, IBM Research Lab, Brazil  
Igor Kotenko, St.Petersburg Institute for Informatics and Automation of the Russian Academy, Russia  
Boris Kovalerchuk, Central Washington University - Ellensburg, USA  
Satoshi Kurihara, University of Osaka, Japan  
Helge Langseth, NTNU, Norway  
Jingpeng Li, The University of Nottingham - Ningbo, China  
Fidel Liberal Malaina, University of the Basque Country, Spain  
Antonio Liotta, Eindhoven University of Technology, The Netherlands  
Hai-Bin Liu, China Aerospace Engineering Consultation Center, China  
Jaime Lloret Mauri, Polytechnic University of Valencia, Spain  
Angela Locoro, University of Genova, Italy  
Noel Lopes, Polytechnic of Guarda, Portugal  
Hanan Lutfiyya, The University of Western Ontario - London, Canada  
Prabhat Mahanti, University of New Brunswick, Canada  
Sayyed Majid Esmailifar, Sharif University of Technology -Tehran, Iran  
Jacques Malenfant, Université Pierre et Marie Curie, France  
Elisa Marengo, Università degli Studi di Torino, Italy  
Mauricio Marin, Universidad de Santiago and Yahoo! Labs Santiago, Chile  
Goreti Marreiros, Polytechnic of Porto, Portugal  
Patrick Martin, Queen's University - Kingston, Canada

Rajat Mehrotra, Mississippi State University - Starkville, USA  
Yasser F. O. Mohammad, Assiut University, Egypt / Kyoto University, Japan  
Thierry Monteil, LAAS-CNRS, INSA de Toulouse, Toulouse, France  
José Moreira, University of Aveiro, Portugal  
Masayuki Murata, Osaka University, Japan  
José Neves, Universidade do Minho - Braga, Portugal  
John O'Donovan, University of California, Santa Barbara, USA  
Andreas Oberweis, Karlsruhe Institute of Technology (KIT), Germany  
Jonice Oliveira, Federal University of Rio de Janeiro, Brazil  
Michael O'Mahony, University College Dublin, Ireland  
Jose Oscar Fajardo, University of the Basque Country, Spain  
David Ostrowski, Ford Motor Company / University of Michigan - Dearborn, USA  
Umberto Panniello, Politecnico di Bari, Italy  
Nandan Parameswaran, University of New South Wales - Sydney, Australia  
Luis Paulo Reis, University of Minho, Portugal  
Loris Penserini, European Commission, Belgium  
Mark Perry, University of New England in Armidale, Australia  
Steve Phelps, University of Essex, UK  
Agostino Poggi, Università degli Studi di Parma, Italy  
Wendy Powley, Queen's University - Kingston, Canada  
Francesco Quaglia, Sapienza Università di Roma, Italy  
Kanagasabai Rajaraman, Institute for Infocomm Research, Singapore  
Alejandro Ramirez-Serrano, University of Calgary - Alberta, Canada  
Martin Randles, Liverpool John Moores University, UK  
Christoph Rasche, University of Paderborn, Germany  
Marek Reformat, University of Alberta, Canada  
Paolo Romano, INESC-ID Lisbon, Portugal  
Rosaldo Rossetti, University of Porto, Portugal  
Cristian Ruz, INRIA Sophia Antipolis Méditerranée, France  
Ricardo Sanz, Universidad Politecnica de Madrid, Spain  
Munehiko Sasajima, Osaka University, Japan  
Andrea Schaerf, DIEGM, University of Udine, Italy  
Christoph Schommer, University Luxemburg, Luxemburg  
Jan Sefranek, Comenius University - Bratislava, Slovakia  
Paulo Jorge Sequeira Gonçalves, Polytechnic Institute of Castelo Branco, Portugal  
Mohamed Shehab, University of North Carolina at Charlotte, USA  
Maxim Shevertalov, Drexel University, USA  
Flavio Soares Correa da Silva, University of Sao Paulo, Brazil  
Edward Stehle, Drexel University, USA  
Claudius Stern, University of Paderborn, Germany  
Ryszard Tadeusiewicz, AGH University of Science and Technology, Poland  
Yuqing Tang, Carnegie Mellon University, USA  
Patricia Tedesco, Federal University of Pernambuco, Brazil  
Michael Tighe, University of Western Ontario - London, Canada  
Manachai Toahchoodee, The University of the Thai Chamber of Commerce, Bangkok, Thailand  
Irina Topalova, Technical University of Sofia, Bulgaria  
José Manuel Torres, University Fernando Pessoa - Porto, Portugal  
Davide Tosi, Università dell'Insubria – Como, Italy

Raquel Trillo Lado, University of Zaragoza, Spain  
Egon L. van den Broek, University of Twente, Enschede, Karakter University Center, Radboud University  
Medical Center Nijmegen, Nijmegen, The Netherlands  
Cristián Varas, NIC Chile Research Labs, Chile  
Eloisa Vargiu, Barcelona Digital Technology Center, Spain  
Phan Cong Vinh, NTT University, Vietnam  
Vladimir Vlassov, KTH Royal Institute of Technology, Sweden  
Stefanos Vrochidis, Centre for Research and Technology Hellas - Themi-Thessaloniki, Greece  
Bozena Wozna-Szczesniak, Jan Dlugosz University, Institute of Mathematics and Computer Science –  
Czestochowa, Poland  
Nanjian Wu, Chinese Academy of Sciences, China  
Reuven Yagel, Ben-Gurion University, Israel  
Constantin-Bala Zamfirescu, "Lucian Blaga" University of Sibiu, Romania  
Andrzej Zbrzezny, University of Warsaw, Poland  
Dieter Zöbel, University Koblenz-Landau, Germany  
Albert Zomaya, University of Sydney, Australia

## Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission to reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.



## Table of Contents

Enabling Ubiquitous Workplace Through Virtualization Technology <i>Carlos Oliveira</i>	1
Towards Autonomous Load Balancing for Mobile Data Stream Processing and Communication Middleware Based on Data Distribution Service <i>Rafael Vasconcelos, Markus Endler, Berto Gomes, and Francisco Silva</i>	7
QoS-Aware Component for Cloud Computing <i>Ines Ayadi, Noemie Simoni, and Gladys Diaz</i>	14
Autonomous Systems: from Requirements to Modeling and Implementation <i>Nikola Serbedzija</i>	21
Comparison Between Self-Stabilizing Clustering Algorithms in Message-Passing Model <i>Mandicou Ba, Olivier Flauzac, Rafik Makhloufi, Florent Nolot, and Ibrahima Niang</i>	27
Goal Processing and Semantic Matchmaking in Opportunistic Activity and Context Recognition Systems <i>Gerold Hoelzl, Marc Kurz, and Alois Ferscha</i>	33
Smart Bin for Incompatible Waste Items <i>Arnab Sinha and Paul Couderc</i>	40
Middleware Supporting Net-Ready Applications for Enhancing Situational Awareness on Rotorcraft <i>Gustavo L. B. Baptista, Markus Endler, Jose Viterbo, Thomas A. DuBois, and Randall L. Johnson</i>	46
A Methodology for Evaluating Complex Interactions between Multiple Autonomic Managers <i>Thaddeus Eze, Richard Anthony, Chris Walshaw, and Alan Soper</i>	53
Organic Programming of Real-Time Operating Systems <i>Lial Khaluf and Franz Josef Rammig</i>	57
Higher Order Sliding Mode Control of Robot Manipulator <i>Awatif Guendouzi, Ahcene Boubakir, and Mustapha Hamerlain</i>	61
Numerical Model of Generalized Multi-Priority Queuing System <i>Eimutis Valakevicius and Mindaugas Snipas</i>	67
HiSPADA: Self-Organising Hierarchies for Large-Scale Multi-Agent Systems <i>Jan-Philipp Steghofer, Pascal Behrmann, Gerrit Anders, Florian Siefert, and Wolfgang Reif</i>	71

Methodology of Training and Support for Urban Search and Rescue With Robots <i>Janusz Bedkowski, Karol Majek, Igor Ostrowski, Pawel Musialik, Andrzej Maslowski, Artur Adamek, Antonio Coelho, and Geert De Cubber</i>	77
Development of Ontology Based Framework for Information Security Standardss <i>Partha Saha, Ambuj Mahanti, Binay Bhushan Chakraborty, and Avinash Navlani</i>	83
Autonomic Diffusive Load Balancing on Many-core Architecture using Simulated Annealing <i>Hyunjik Song and Kiyounng Choi</i>	90
Autonomic Ontologies for Governmental Knowledge Base <i>Stainam Nogueira Brandao, Sergio Assis Rodrigues, Tiago Silva, Luis Araujo, and Jano Moreira de Souza</i>	96

# Enabling Ubiquitous Workplace Through Virtualization Technology

Carlos Roberto de Oliveira Junior

Department of Informatics  
 Instituto Federal de Educação, Ciência e Tecnologia (IFRJ)  
 Rio de Janeiro, Brazil  
 Carlos.roberto@ifrj.edu.br

**Abstract**—In a ubiquitous computing world, users would always have the right information, in the right form, within the right context. In addition, they would like to manipulate their information with their preferred tools. In this paper, an approach based on the advances in virtualization that can change state of art in ubiquitous computing is proposed. Also, how this change can be achieved, showing what has been done and the open issues, is explained. The goal of this paper is not to solve these issues but to give an insight on how the Virtual Machine (VMs) technology can be used towards materializing the ubiquitous computing dream. We concluded that, in a near future, it will be possible to enable ubiquitous computing.

*Keywords*-virtualization; ubiquitous; workplace.

## I. INTRODUCTION

Computers prices are getting down each year, so a person can have more than one machine (PC, notebook, tablet, etc.) [1]. In addition, nowadays many people use another computer at work. This leads to different information and applications in these many computers. Nevertheless, it is desirable to have the same information in all computers.

According to Oulasvirta and Sumari [2] there are 3 main problems while having multiple machines: 1) the physical effort demanded by various management tasks, 2) anticipating what data or functionality will be needed and shared, and 3) to align these efforts with work, mobility, and social situations.

The ability to have data anytime/anywhere can also be useful for companies as they can have their applications servers (running in VMs) “moved” to anywhere at anytime. Having everything anytime/anywhere is an old dream. The term *ubiquitous computing* has been used first in 1988 by Mark Weiser who published the paper *The Computer for the 21st Century* [3] in 1991. To achieve this dream, many problems needs to be solved. Instead of have isolated computers we need to have them integrated in a transparent way to the user; it can be called as a *user interface* problem.

We need also to have in mind that to achieve gracefully this desirable ubiquitous computing we need to integrate all computers and not only propose a solution to integrate a new kind of computer. It is necessary to regard existing ones.

The concept of *ubiquitous workplace*, introduced in this paper, is a service that allows users to have their data anytime/anywhere in a transparent way. In this service, user's data is stored in the cloud. This data is his preferred OS and configured applications. Users can use any computer to

access their data. At the moment, an user logs in a computer, his data would be downloaded to this computer. In our proposal, user's data is represented by a VM and each user has a VM.

Using VMs is possible to carry a whole OS and user's profile from a physical machine to another machine in the same network in a few seconds and even without stopping services running on these VMs [4]. It is also possible to carry VMs across Wide Area Network (WAN) [5]. Moving a VM to anywhere is important because the user can carry all his profile doing it. This is better than moving only files because there are files that need specific applications to be opened. It is also possible to stop tasks in a machine and resume it in a different machine. This is an important feature because it solves the backup problem. Having all profile in a VM, the user needs not to worry about doing backups because all changes are directly applied to the VM. Moreover, the user needs not to worry about reinstall a system when buying a new machine. Running the VM in the new machine, the user will be carrying the whole system to the new computer.

Our purpose is not to solve problems in ubiquitous computing, but we intend to propose a new manner to advance the state of art using VM technology, showing what has been done and the open problems. We introduce the concept of *ubiquitous workplace*. An ubiquitous workplace consists of a VM that can be accessed independent on the physical machine the user is current using. This VM contains any OS and user applications, like it would be if the user was using an OS installed directly on a physical machine. Our goal is to allow the user to access his whole profile with all his data and configured applications anytime/anywhere without worry about the physical machine current being used. The machine is only a way to access the ubiquitous workplace.

The proposed architecture consists of three main agents: clients, client's file system and VM repositories. Clients are ubiquitous workplaces users. A client uses any computer to access his ubiquitous workplace. Clients need not to worry about the underlying OS, installed in the machine currently being used. A prerequisite is that computers need to have a virtual machine monitor (VMM) installed to allow clients to run VMs. Clients VM's are stored in a VM repository. This repository can be a cloud service like Amazon EC2 [6]. Client's VM are downloaded to the machine the client is currently accessing at the time he logs in. When a client

finishes his work, his VM is updated to the repository. Later on, his VM can be migrated to another machine.

In this paper, we discuss five requirements that must be addressed and integrated in an unique support to allow a complete implementation of ubiquitous workplace. In the future, when we have these five requirements addressed, it will be possible to implement an autonomous system in which the user workplace is automatically moved from a physical machine to the physical machine the user is currently using. In previous works [42] [43], we have already implemented an autonomous system in which VMs (containing web servers) are moved from a physical machine to another according to resource usage constraints. This implementation could be adapted to manage the user workplaces.

The rest of the paper is organized as follows. Section II describes the basic requirements for the implementation of our proposal. Section III points out some scenarios in which our proposal could improve user experience with an ubiquitous workplace. Section IV presents related work and Section V concludes the paper.

## II. REQUIREMENTS

Our proposal is based on virtualization that is a consolidated technique. But some essential issues and their related requirements need to be addressed in order to allow a complete implementation of our proposal. In the next Subsections we present these requirements and discuss the solutions proposed by researchers to address each of them.

### A. Creating a VM from an installed system

In our proposal, users need to have a VM. But we need to take into account those users that do not use virtualization and have already their systems installed in a physical machine and their profiles, files and applications installed and configured on the physical machine. We cannot propose to users to leave their systems and to start a new installation into a VM. Thus, the first requirement is to allow users to create their VMs starting from their installed systems. To do this, we need to have a service that allows users to submit their systems and the service returns a VM containing this system.

Some tools like VMware Converter [7], that is a free product, provide this service. The process to create a VM, using products like this is (a) The user has an installed OS (note that an user can have any OS) containing his applications. (b) User uses an application to create a VM containing his system. (c) The application creates a VM with his whole system. This VM can run in any computer that supports the VMM the VM was created to, even if the VMM are installed over a different OS. For instance, suppose that OS in user's VM is Windows. This VM can run in a computer that has the same VMM installed over Linux.

An issue here is that, sometimes, the OS the VMM is running over does not offer all the capabilities the guest OS offer. Thus, the user can face some troubles when trying to use a capability that is not supported by the host OS.

### B. Migrating VMs in WANs

As we want an ubiquitous workplace, we need to be able to move VMs to outside of LANs. Migrating a VM in WANs have been studied and some authors have published their work and contributions.

Bradford et al. [5] have studied VMs migration in WANs. Their contribution is the design and evaluation of a system that enables live migration of VMs that (a) uses local storage, and (b) have open network connections, without severely disrupting their live services, even across the Internet.

When the migration is in a LAN, the VM's image (its file system) is not moved from the source host to the target host. The VM's image is stored in a Storage Area Network (SAN) or in a Network File System (NFS). Thus, only the memory state needs to be migrated. However, as we want to access a VM anywhere we cannot be limited to use SAN or NFS. Bradford et al. [5] use Xen as VMM and they use *block tap* [8] to export block devices. Their system pre-copies local persistent state and transfers it from the source to the destination while the VM operates on the source host. During the transfer it is employed an user-level block device to record and forward any write accesses to the destination, to ensure consistency. In their experiments, they have complete a migration in the WAN in 68s. Two related issues in the migration context are discussed below.

#### 1) Keeping open connections

There are a few number of scenarios in which should be necessary to keep open connections (e.g., when the user's VM is migrated to the cloud with an ongoing download). As described in the following paragraphs, it is possible to handle these situations in the context of an ubiquitous workplace.

For managing migration in a LAN with respect to network it is generated an unsolicited ARP reply from the migrated host, advertising that the IP has moved to a new location. This will reconfigure peers to send packets to the new physical address and, while a small number of in-flight packets may be lost, the migrated domain will be able to continue using open connections with almost no observable interference. Some routers are configured not to accept broadcast ARP replies (in order to prevent IP spoofing), so an unsolicited ARP may not work in all scenarios [4].

When migration takes place between servers in different networks, the migrated VM has to obtain a new IP address and thus existing network connections break. Bradford et al. [5] implemented a redirection scheme in their framework to overcome this by combining IP tunneling [11] with Dynamic DNS [10]. With the help of *iproute2* they set up an IP tunnel between the old IP address at the source and its new IP at the destination. Once the migration has completed and the VM can respond at its new network location they update the Dynamic DNS entry for the services the VM provides. This ensures that future connections are directed to the VM new IP address. In addition, they begin forwarding all packets that arrive at the source for the VM old IP address to the destination, through the tunnel. Packets that arrive during the final migration step have to either be dropped or queued in

order to avoid connection resets. After the restart of the VM at the destination the VM has two IP addresses: its old one, used by existing connections through the tunnel, and its new one, used by new connections directly. The tunnel is torn down when no connections remain that use the VM old IP address.

## 2) *Discovering and mapping devices*

We need not only to keep open connections but we also need to discover available devices where the user currently is. An example that shows the importance of discovering and mapping devices is an user who has a printer configured at home. But when the user moves to the office, that configured printer is not available anymore. Thus, we need to discover available printers at the new location and configure them in a transparent way (when it is possible) to the user. We need also to consider that devices may be added or removed anytime. To discover available devices we can use an approach similar to Gaia [11], which is a framework for smart spaces. Gaia consists of a number of different types of agents performing different tasks. There are agents that perform various core services required for the functioning of the environment like device discovery and selection. There are agents associated with devices that enable them to be a part of a particular environment where they can be readily available [12]. We must take in account that should be difficult to implement a system to discover and map devices in an environment where the user is continuously moving. The system can experiment problems like different passwords and authentication methods.

## C. *VM storage repository*

As we are proposing the use of VM also for end-users and not only for enterprises, we should have a way to store VMs in an accessible manner for these users. When a company uses virtualization, it relies on NFS or SAN support to store the VMs. When thinking about end-users it is a bit different. These users will likely not need to have their VMs running all the time but probably only when they are working in front of a computer. And so, they probably do not want to leave their machines at home turned on all the time to make possible to migrate their VMs when finishing their works on the machine they use at the office, for example. Thus, the user has two options: (a) leave his machine in the office turned on and when arriving at home remotely accesses his computer at the office to migrate his VM or (b) have another place to store his VM as he cannot leave his machine turned on. The last option sounds better.

With the advance of cloud computing, our proposal can use a popular service like Amazon EC2 [6] to store the VMs and let it accessible anytime/anywhere. Users can have their VMs updated to the cloud when they need to go away, log off, sleep, etc and have their VMs downloaded when necessary. The idea is that after an user stops working, his VM is migrated to the cloud and still there ready to be used when necessary. Other proposals have been done in this area. For instance, a storage access mechanism that supports live VM migration over WAN is proposed in [13]. This

mechanism completely relocates virtual disks between source and remote sites in a transparent manner.

An important issue in this requirement is that a VM (that is, the OS and applications) can have dozens of gigabytes. This large amount of data can turn unfeasible the task to move a whole VM through the network. An approach to address this issue is to transfer only the OS image to the machine the user is currently using, while user's file system (that is, his files) stays in the cloud. Memory pages from the file system storage are transferred on demand to the machine the user is using. It would work as a NFS and will decrease the amount of data transferred through the Internet. It could also be used a file system like Coda [14]. In Coda, Kistler and Satyanarayanan suggest a disconnected operation mode that enables a client to continue accessing critical data during temporary failures of a shared data repository. Their work can be seen as an extension of the idea of write-back caching. Whereas write-back caching has been used for performance, it can be extended to mask temporary failures too.

## D. *Independence of VMM*

To advance in the state of art we cannot propose a solution that uses a specific VMM. We cannot force users to use what we want them to use, they must be free to choose their preferred VMM. Ideally, a VM could be migrated across any node with similar machine configuration and granted resources. However, as there are currently several VMMs (e.g, Denali [15], VMware [16], Xen [17] and kvm [18]), the heterogeneity of the underlying VMMs makes it hard to migrate a VM originally running on one VMM to a system that runs another VMM. To address this problem, authors in [19] have proposed and implemented Vagrant, which is a VM migration scheme aiming to migrate VMs among computing nodes even if they are managed by heterogeneous VMMs. To render it practical, Vagrant does not requires changes to the hosted OS and its applications. It requires only minimal changes to the core VMM abstraction.

To support migration among heterogeneous VMMs, the key issue is the semantic gap among different resource abstractions and migration protocols. First, each VMM provides its own abstraction of the underlying hardware resources. Second, the migration software for different VMMs usually has its own format of migration protocols. Third, there are currently several memory migrating algorithms, including stop-and-copy [8], pre-copy [4], push and pull. To address this issue, Vagrant has a common migration protocol and common virtual machine abstraction. It intercepts the migration control and data issued by the source VMM and transforms them into the Vagrant format. On the destination side, the data in Vagrant format is transformed into the format of the target VMM. For memory migration algorithms, Vagrant provides a pool of common algorithms in both VMMs and dynamically selects the migration algorithms according to the types of the communicating VMMs. Until now, Vagrant supports only migration within LAN. Nevertheless, its authors plan to extend it to support migration in WAN. They also plan to

add features such as QoS control to make Vagrant more flexible and robust. Using mechanisms like this, we expect that, in the future, it will be practical to migrate VMs without worrying about the particular available VMM.

#### E. High speed and reliable Internet access

Nowadays, pen drives can store many gigabytes; thus, an user could carry his VM anywhere using his pen drive. But this option is not convenient or comfortable to the user, and it is against the idea of pervasive computing. Thus, we need to have a high speed and reliable Internet access, allowing us to access our VMs in a transparent way. Many companies and universities have gigabit links to access Internet, but we need to have a high speed access to domestic users too. Moreover, we need to have a reliable Internet access. Users must be able to access their VMs anytime. The American government proposed a National Broadband Plan that is an initiative to bring 100-megabit-per-second broadband to 100 million U.S. households by 2020 [20]. This service is already reality in Japan that has the fastest consumer broadband in the world, 160-megabit-per-second service [21]. In Japan, an user should pay only \$20 to have a service like this at home.

As exposed in this Section, all the five essential requirements we have identified were successfully addressed by researchers. Nevertheless, they still need to be integrated in a simple solution to users who are not acquainted with the virtualization technology.

### III. ADVANCES EXPECTED IN OUR LIVES

In this Section, we show a scenario that would be possible after integrate these five requirements. Consider the following scenario.

*“An user leaves office and afterwards continue working at home”*

During the day at office one can create or modify documents, receive or modify files, install and uninstall applications and would like to have these changes applied to his computer at home. Moreover, imagine a student who has a work in progress. This student can make some progress in his work at university and may want to continue this work at home. This work may be complex, for instance, requiring a specific system configuration. Using a ubiquitous workplace it is possible to continue his work anywhere.

A further advantage of our proposal is that companies and universities do not need to worry about untrustful user's applications. In a virtualized system, a VM runs independently from others. It leads immediately to a safe environment. For instance, if an user has a virus on his VM, this virus cannot be propagated to other VMs because they are isolated from each other.

Taking into account requirement 1, one may have his VM. By requirements 2, 3 and 5, one can store and access

his VM with all his configured environment anytime. We need also to consider that the company or university may have a payed VMM while a employee or student has a free VMM at home. Thus, by addressing requirement 4, an user can move his ubiquitous workplace to anywhere. That is, from a computer to another computer independently of the VMM used on each computer.

### IV. RELATED WORK

Remote Desktop [30], included with Windows XP Professional, enables users to connect to their computers across the Internet from any computer. The disadvantage of Remote Desktop is that users need to leave their computers turned on all the time. Moreover, this computer is a single point of failure and the user has no choice about the OS.

Windows 7 [31] brings a new feature termed HomeGroup [32] that allows users to have all computers at home integrated. It allows users to use a machine to access a file stored in another machine. For instance, one can view a picture that is stored in another physical machine on his HomeGroup. One can also use a printer and other devices that are connected to other machine. Is even possible to give permissions to files, specifying who can access which files. The disadvantage of HomeGroup is that if the user goes to the office, for instance, he cannot access his files as he could do at home. HomeGroup can be used only for machines in the same network. So, it does not allow users to access their data anytime/anywhere.

Another proposed solution to allow users to access data anytime/anywhere were the Web OSs. One of these Web OS is Jooce [33]. Jooce is a Flash-based web OS and sharing platform. Google has also its proposal to a Web OS. It is termed ChromeOS [34], that aims to allow users to quickly access Internet and have their applications and data stored on the Internet. They start from the assumption that when users are using a computer, they spend 90% of time on the Internet. In our view, Web OSs were not well succeed mainly because they can run only applications that have been written for them. Simply, users do not want to leave their OSs and applications to change to a completely new OS.

f\_Desktop [35] is another proposal that aims to support user mobility in ubiquitous computing environment. f\_Desktop allows the user to carry his desktop computing environment defined as a set of follow-me applications from computer to computer. f\_Desktop uses MobileSocket to migrate applications. Nevertheless, f\_Desktop is based on Java, so it is only possible to use applications based on Java.

Intel has proposed the *Personal Server* [36] as a solution to allow users to have data anywhere. Its proposal aims to allow users to carry their data anywhere. *Personal Server* is a device that has no screen. The Personal Server aims to overcome the fundamental limitation of cell-phones, PDAs, and laptops: if they're small enough to carry, then the displays are too small to easily use. Thus, *Personal Server* needs to be connected to another device that has a screen.

The Internet Suspend/Resume [37] is a project that aims to provide mobile computing by combining two technologies: VM technology and distributed file systems.

As far as we know, this was the first work to propose the use of VM technology to allow mobile computing; it demonstrates that the basic support required by our proposal is feasible. Some issues not considered in [37] are explicitly discussed in our work.

Das et al. [38] implemented a system, termed LiteGreen that aims to save desktop energy by virtualizing the user's desktop computing environment as a virtual machine (VM) and then migrating it between the user's physical desktop machine and a VM server, depending on whether the desktop computing environment is being actively used or is idle. This operation allows LiteGreen to save energy during short idle periods (e.g., coffee breaks) and long idle periods (e.g., weekends). In [38], the ubiquitous computing was not achieved as the system is limited to be used in an enterprise environment. As a future work, we plan to implement a system that, similarly to LiteGreen, moves the VM between a cloud service and user's PC whether the user logs in/out a web site that provides our service. Unlike LiteGreen, our proposal requires no changes to the desktop. In our paradigm, is not desirable or convenient to apply modifications to desktops, as the user should be able to use any desktop.

## V. DISCUSSION AND CONCLUSION

System virtualization has become a disrupting force in the computer systems and is likely to be the new foundation of system software [39]. A conspicuous feature enabled by system virtualization is the migration of VMs. In principle, by dynamically relocating an entire environment including the operating system from one machine to another, we can achieve the ubiquitous workplace paradigm. The five requirements to support our proposal have already been addressed by many researchers as presented in Section 2. The next step is to integrate these requirements in a comprehensive environment and make it available to be used by people who do not want to know about the underlying technology.

Many proposals with similar goals in ubiquitous computing and related areas have failed mainly because they tried to impose to users their specific solutions or tools. Now, we know that if we want to change the state of art we must propose a feasible solution to the general users and not only for a specific group. In particular, it is not fair trying to convince them or impose to them a particular choice (as Web OS tried to do). In sum, giving freedom to the user to choose his preferred environment is essential. It is described below some benefits of our proposal.

- Access to the ubiquitous workplace anytime/anywhere: an user can do all his work on his ubiquitous workplace using a generic machine and when he needs to go to another place he can easily continue his work on another different machine. As a nice effect users would not need to worry about migrating to a new hardware platform, if by any reason they decide to change the machine they keep at their homes

or offices. In the future, when the five identified requirements are fully integrated, users will not need to worry about particular computers because they will be widely available. They would only need a way to connect to their ubiquitous workplace.

- Collaborative work: an user can send a workplace to a colleague in another place to continue or evaluate his work. For instance, a workgroup team can have a developer working in a country and the tester working in another country. Instead of sending only the code to the person who will test it (who can have to reconfigure his system to run the application), the developer can send a copy of his workplace with the whole associated environment.
- Data sharing: taking advantage of the cloud-based storage, users could share data (photos, videos, etc) with friends or familiars just sharing their local repositories. This would allow the partners to use their preferred tools (e.g., viewers). This would avoid users the stress of having to download, install and learn to use a specific viewer to see the shared media.
- Backup problem: all modifications the user does in the environment (e.g., by installing programs or modifying data) are immediately applied to the underlying VM, that at any given moment can be reflected in the cloud-based storage.
- Energy savings: as users tend to create huge amounts of data during their lives, by keeping the currently unused part of this data in the cloud can be useful to save energy, as pointed out in [39].
- User habits preserved: many proposals until now were based on new abstractions forcing the user to change his or hers habits. Our proposal is practical and feasible because virtualization allows one to have only one ubiquitous workplace, containing his preferences, programs and data, accessible anytime/anywhere.

We also expect that in the future, vendors will sell machines with the basic virtualization mechanisms required to implement our proposal. Moreover, ubiquitous workplaces will be easily stored in cloud services and users will have high speed and reliable access to them through the Internet. Our expectations are in accordance with Google [40] and IBM [41] that foresee that desktop concept will be obsolete in a few years.

## REFERENCES

- [1] D. Dearman, and J. S. Pierce, "It's on my other Computer!: Computing with Multiple Devices," Conference on Human Factors in Computing Systems, 2008, pp. 767-776.

- [2] A. Oulasvirta, and L. Sumari, "Mobile kits and laptop trays: Managing multiple devices in mobile information work," ACM, 2007, pp. 1127-1136.
- [3] M. Weiser, "The computer for the 21st century," Scientific American, 1991, pp. 94-104.
- [4] C. Clark et al, "Live migration of virtual machines," IEEE, 2005, pp. 273-286.
- [5] R. Bradford et al, "Livewide-area migration of virtual machines including local persistent state," ACM/Usenix International Conference On Virtual Execution Environments, 2007, pp. 169-179.
- [6] Amazon, "Amazon elastic compute cloud", <http://aws.amazon.com/ec2/> [retrieved: 02, 2013]
- [7] VMware, "Vmware vcenter converter", <http://www.vmware.com/products/converter/> [retrieved: 02, 2013]
- [8] A. Warfield, H. Steven, K. Fraser and T. Deegan, "Facilitating the development of soft devices," USENIX Annual Technical Conference, 2005, pp. 379-382.
- [9] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam and M. Rosenblum, "Optimizing the migration of virtual computers," in ACM SIGOPS Operating Systems Review, 2002, pp. 377 - 390.
- [10] B. Wellington, "Secure dns dynamic update," RFC 3007, 2000.
- [11] M. Rom'an and R. H. Campbell, "A middleware-based application framework for active space applications," in Middleware '03: Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware. New York, NY, USA: Springer-Verlag New York, Inc., 2003, pp. 433-454.
- [12] A. Ranganathan and R. H. Campbell, "A middleware for context-aware agents in ubiquitous computing environments," in Middleware '03: Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware. New York, NY, USA: Springer-Verlag New York, Inc., 2003, pp. 143-161.
- [13] T. Hirofuchi, H. Nakada, H. Ogawa, S. Itoh, and S. Sekiguchi, "A live storage migration mechanism over wan and its performance evaluation," in VTDC '09: Proceedings of the 3rd international workshop on Virtualization technologies in distributed computing. New York, NY, USA: ACM, 2009, pp. 67-74.
- [14] J. J. Kistler and M. Satyanarayanan, "Disconnected operation in the coda file system," ACM Trans. Comput. Syst., vol. 10, no. 1, pp. 3-25, 1992.
- [15] A. Whitaker, M. Shaw and S. D. Gribble., "Scale and performance in the denali isolation kernel," TACM SIGOPS Operating Systems, 2002, pp. 195-209.
- [16] J. Sugeran, G. Venkitachalam and B. Lim, "Virtualizing i/o devices on vmware workstation's hosted virtual machine monitor," USENIX Annual Technical Conference, 2001, pp. 1-14.
- [17] P. Barham et al., "Xen and the art of virtualization," in SOSP'03. ACM, 2003, pp. 164-177.
- [18] A. K. Qumranet, "kvm: the linux virtual machine monitor," in Proceedings of the Linux Symposium, 2007, pp. 225-230.
- [19] P. Liu, Z. Yang, X. Song, Y. Zhou, H. Chen and B. Zang, "Heterogeneous live migration of virtual machines," International Symposium on Computer Architecture, 2008.
- [20] PC World, "Fcc's 100 megabits to the home: What it means to you," <http://www.pcworld.com/article/189507/>. [retrieved: 02, 2013]
- [21] The New Yor Times, "Worldest fastest broadband at \$20 per home," <http://bits.blogs.nytimes.com/2009/04/03/>. [retrieved: 02, 2013]
- [22] VMware, "Vmware workstation: The gold standard in desktop virtualization," <http://www.vmware.com/products/workstation/> [retrieved: 02, 2013]
- [23] Microsoft, "Windows xp," <http://www.microsoft.com/windows/windows-xp/> [retrieved: 02, 2013]
- [24] Microsoft, "Windows xp service pack 3 overview," <http://www.microsoft.com/downloads/details.aspx?FamilyId=68C48DAD-BC34-40BE-8D85-6BB4F56F5110n&displaylang=en> [retrieved: 02, 2013]
- [25] Microsoft, "Windows media," <http://www.microsoft.com/windows/windowsmedia/> [retrieved: 02, 2013]
- [26] Microsoft, "Internet explorer," <http://www.microsoft.com/windows/internet-explorer/default.aspx> [retrieved: 02, 2013]
- [27] Microsoft, "Communicate instantly with windows messenger," <http://www.microsoft.com/windowsxp/using/windowsmessenger/> [retrieved: 02, 2013]
- [28] Eucalyptus Systems, "Eucalyptus systems," <http://www.eucalyptus.com/> [retrieved: 02, 2013]
- [29] VMware, "Vmware vsphere hypervisor (esxi): Virtualization made free and easy," <http://www.vmware.com/products/vsphere-hypervisor/> [retrieved: 02, 2013]
- [30] Microsoft, "Get started using remote desktop with windows xp professional," <http://www.microsoft.com/windowsxp/using/mobility/getstarted/remo teintro.mspx> [retrieved: 02, 2013]
- [31] Microsoft, "Windows 7 features," <http://www.microsoft.com/windows/windows-7/default.aspx> [retrieved: 02, 2013]
- [32] Microsoft, "Windows 7 features: Homegroup," <http://www.microsoft.com/windows/windows-7/features/homegroup.aspx> [retrieved: 02, 2013]
- [33] Jooce, "Jooce," <http://jooce.com/> [retrieved: 02, 2013]
- [34] Google, "Introducing the google chrome os," <http://googleblog.blogspot.com/2009/07/introducing-google-chrome-os.html> [retrieved: 02, 2013]
- [35] K. Takashio, G. Soeda and H. Tokuda, "A mobile agent framework for follow-me applications in ubiquitous computing environment," in ICDCSW'01: Proceedings of the 21st International Conference on Distributed Computing Systems. Washington, DC, USA: IEEE Computer Society, 2001, p. 202-207.
- [36] R.Want, T. Pering, G. Danneels, M. Kumar, M. Sundar and J. Light, "The personal server: Changing the way we think about ubiquitous computing," in ACM International Conference on Ubiquitous Computing. Goteborg, Sweden: IEEE Computer Society, 2002, pp. 194-209.
- [37] M. Kozuch and M. Satyanarayanan, "Internet suspend/resume," in WMCSA '02: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications. Washington, DC, USA: IEEE Computer Society, 2002, p. 40.
- [38] T. Das, P. Padala, V. N. Padmanabhan, R. Ramjee, and K. G. Shin, "Litegreen: saving energy in networked desktops using virtualization," in USENIXATC'10: Proceedings of the 2010 USENIX conference on USENIX annual technical conference. Berkeley, CA, USA: USENIX Association, 2010, pp. 3-3.
- [39] S. Crosby and D. Brown, "The virtualization reality," in ACM Q focus: computer architecture, 2006, pp. 34 - 41.
- [40] Google, "Google exec: Desktops will be obsolete in 3 years," [http://www.maximumpc.com/article/news/google\\_exec\\_desktops\\_will\\_be\\_obsolete\\_3\\_years](http://www.maximumpc.com/article/news/google_exec_desktops_will_be_obsolete_3_years) [retrieved: 02, 2013]
- [41] Computerworld UK, "Ibm says desktop computers and phones will disappear," <http://www.computerworlduk.com/technology/networking/voip/news-analysis/index.cfm?articleid=1273> [retrieved: 02, 2013]
- [42] C. Oliveira, "Load balancing on virtualized web servers," in SRMPDS '12: Proceedings of the Eighth International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems. Pittsburgh, USA.
- [43] C. Oliveira, "An Architecture for Experimentation with Multi-tier Internet Applications in Virtualized Server Environments," in WoSIDA'12: Proceedings of the Second Workshop on Autonomic Distributed Systems. Ouro Preto, Brazil: Brazilian Computer Society.



## Towards Autonomous Load Balancing for Mobile Data Stream Processing and Communication Middleware Based on Data Distribution Service

Rafael Oliveira Vasconcelos and Markus Endler  
*Department of Informatics*  
*Pontifical Catholic University of Rio de Janeiro (PUC-Rio)*  
*Rio de Janeiro, Brazil*  
*rvasconcelos@inf.puc-rio.br, endler@inf.puc-rio.br*

Berto de T. P. Gomes and Francisco J. da Silva e Silva  
*Graduate Program Electric Engineering (PPGEE)*  
*Federal University of Maranhão (UFMA)*  
*São Luís, Brazil*  
*bertodetacio@ifma.edu.br, fssilva@deinf.ufma.br*

**Abstract**—Intelligent Transportation Systems, Fleet Management and Logistics, and integrated Industrial Process Automation share the requirement of remote monitoring and high performance processing of huge data streams produced by large sets of mobile nodes. The deployment and operation of infra-structures enabling such mobile communication and data stream processing have two key requirements: they must be capable of (a) handling large and variable numbers of wireless connections to the monitored mobile nodes regardless of their current use or locations, and (b) automatically adapt to variations in the volume of the mobile data streams. This article describes a mechanism for load balancing of mobile data streams based on the autonomic reference model MAPE-K. The autonomic capability has been incorporated into a scalable middleware system based on the OMG DDS standard and aimed at real-time and adaptive handling of mobile connectivity and data stream processing for great sets of mobile nodes. Besides explaining the autonomic extension of this middleware (MAPE-SDDL) and the generic load balancing approach implemented, we also present results of initial tests and discuss the potential benefits of using this model for general dynamic adaptivity in distributed middleware.

**Keywords**—Load balancing; Data Stream Processing; Autonomic computing; DDS; mobile communication middleware.

### I. INTRODUCTION

As the public and private sectors are increasingly investing in Intelligent Transportation Systems for smarter cities, Logistics, and integrated Industrial Process Automation that depend on monitoring of the real world objects (and people) through wireless interfaces, systems for mobile tracking, communication and coordination are becoming commonplace. Such applications share the requirement of remote monitoring and high performance processing of huge data streams produced by large sets of mobile nodes, which may be vehicles, people, or other smart objects with embedded sensors. Although some sorts of data processing may be performed locally at the mobile nodes (i.e., simple sensor data encoding or interpretation), most other application-relevant information about the monitored mobile system as a whole that requires execution of complex analysis and correlation functions over these mobile data streams, in real-time, has to be done by dedicated machines in a cluster

or cloud. For example, it may be necessary to identify all the nodes that are located inside a region affected by some problem/accident and which may imply some danger or cost to the corresponding nearby mobile users. Moreover, these systems must also support timely and reliable communication with the monitored mobile nodes, in order to send instructions, share alternative routes, make enquiries or disseminate alerts, either to nodes individually, or to groups of nodes [1].

However, today's mobile communication and data stream processing systems lack important features that are necessary in order to support the large amounts of data flows envisioned by the massive and ubiquitous dissemination of sensors and mobile devices in industry and city-scale applications. In particular, the deployment and 24x7 operation of such mobile data stream processing and communication infra-structures pose two intrinsic technical challenges: they must be capable of (a) handling huge and variable numbers of connections to the monitored mobile nodes regardless of their current locations, and (b) automatically adapt to variations in the volume of the mobile data streams, either because of sudden increase in the set of nodes (e.g. a popular event happening in a region), intensified message exchange among the mobiles, or because new sensed data is required for a more precise analysis of a regional problem, such as road defect or an potential accident. In order to address these challenges we have developed a scalable middleware system that supports efficient and adaptive handling of mobile connectivity and data stream processing for thousands of mobile nodes. In this paper, we specifically explain the autonomic load distribution mechanisms implemented in the middleware, and discuss their potential benefits.

The remainder of this paper is organized as follows: Section II gives an overview of the enabling technologies, which are the Data Distribution Service for Real-Time Systems standard and the MAPE-K reference model for autonomic systems; Section III presents the Scalable Data Distribution Layer (SDDL) used as the middleware for mobile communications and the MAPE-SDDL extension, which adds autonomic capabilities to the SDDL middleware;

Section IV delves into the proposed *Data Processing Slice Load Balancing* approach for mobile data streams and shows initial performance results of the implemented system using a prototype application. Section V reviews related work on load balancing for Publish/Subscribe systems, including DDS. Section VI discusses the advantages of using an autonomic approach for load balancing and the benefits of the load balancing solution proposed by this work. Section VII contains with some concluding remarks about the central ideas presented in this work and with probable lines of future work on the subject.

## II. ENABLING TECHNOLOGIES

The efficiency, scalability and adaptiveness of our middleware system builds upon a combination of the following two key technologies.

### A. Data Distribution Service for Real-Time Systems

The Data Distribution Service for Real-Time Systems (DDS) [2] is an Object Management Group (OMG) standard which specifies a publish/subscribe communication infrastructure aimed at high performance and real-time distribution of critical information in distributed systems. This specification was designed with the intention of obtaining a high scalability, portability and interoperability [3]. The DDS was conceived around a Data-Centric Publish-Subscribe (DCPS) model based on topics, which makes the complex programming of distributed communication protocols transparent to the developer. Topics are typed structures that connect Publishers to Subscribers and is where it is located the information that will be exchanged on the network. Publishers and Subscribers of a DDS Domain (the collection of nodes pertaining to a single application) are containers for Data Writers and Data Readers, respectively, which exchange typed data through a common Topic.

Specifically, the DCPS automatically manages delivery of all DDS messages in a totally decoupled and asynchronous way, i.e. without requiring the application to explicitly determine which will be the sender and receiver of each message, or handle message acknowledgements and retransmissions. The DCPS makes it possible to organize its Topics in a relational model, providing support for identity and relations, i.e. for each Topic it is possible to define one or more primary keys, and any number of foreign keys representing, respectively, relationships with other Topics. It also supports a large array of Quality of Service (QoS) policies for communication (e.g. best effort, reliable, ownership, several levels of data persistency, data flow prioritization and several other message delivery options) [3] [4].

Unlike traditional Publish-Subscribe middleware, DDS can explicitly control the latency and efficient use of network resources through fine-tuning of its Network Services, that are critical for implementing real-time and soft real-time

systems that use QoS policies such as Deadline, Latency Budget, Transport Priority, etc.

Among the several existing implementations, we chose CoreDX DDS [5] as the basic communication substrate. The reason for choosing DDS as the communication layer is the fact that it is focused on high-performance and low-overhead, with great latency and message throughput numbers. CoreDX DDS includes fundamental design principles aimed to meet the requirements of real-time and near-real time systems, including minimal data copies, compact encoding on the wire, light-weight notification mechanisms, pre-allocation of resources and pre-compilation of type-specific code blocs.

### B. MAPE-K: a Reference Model for Autonomic Systems

The MAPE-K [6] (*Monitoring, Analysis, Planning, Executing and Knowledge*) model, illustrated in Figure 1, is a general architecture for the development of autonomic software components, as proposed by IBM [7]. This model is being increasingly used to interrelate the architectural components of autonomic systems. It is divided into two main components: autonomic manager and managed resource.

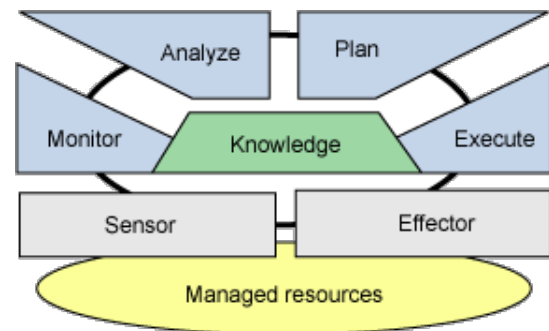


Figure 1. MAPE-K Autonomic Architecture

The managed resource corresponds to the system or some system component providing the business logic that is to be dynamically adapted as the computing environment changes. The managed resource can be, for instance, a Web server, a database, a software component in a given application (e.g., the query optimizer in a database), an operating system, etc. The autonomic manager performs all the functions comprising the adaptation logic on the managed resource: monitoring, analysis, planning, and adaptation execution. This model requires two types of touch points within and outside the managed resource: sensors and effectors. Only they have direct access to the managed resource. Sensors are responsible for collecting information from the managed resource, which can be, for instance, the customers requests response time, if the managed resource is a Web server. The information collected by the sensors are reach the monitors, where they are interpreted, classified and placed in a higher level of abstraction. They are then sent to the

next step of the cycle, the analysis and planning phase. This stage produces an action plan, which consists of a set of adaptation actions to be performed by the executor. The effectors are the components that allow the autonomic managers to perform adjustments in the managed resources. The decision of which adaptation actions must be applied in a given situation requires a knowledge representation of the computing system and its environment. This knowledge can be represented and processed in different ways (e.g., Ontologies, basic ECA-Rules, machine learning, etc.) and must be shared among the monitoring, analysis, planning and executing services of the autonomic manager.

### III. THE SDDL MIDDLEWARE AND ITS EXTENSIONS

#### A. Overview of the Scalable Data Distribution Layer

The Scalable Data Distribution Layer (SDDL) [4] [8] is a communication middleware that connects stationary nodes running in a DDS Domain and deployed in a cloud to mobile nodes that have an IP-based wireless data connection, as illustrated in Figure 2. Some of the stationary nodes are data stream processing nodes, while others are gateways for the communication with the mobile nodes (MNs). Gateways use the Mobile Reliable UDP (MR-UDP) protocol to maintain a virtual connection with each MN. The MR-UDP protocol was developed to be robust to short-lived wireless disconnections, IP address changes of the MNs and capable of Firewall/NAT traversal. One of the nodes in the DDS Domain, the Controller, is also a Web Server that can be accessed by a Web browser, for displaying all the MN's current position (or any other node specific information) and for send unicast, broadcast, or groupcast message to the mobile nodes. Figure 2 shows other nodes in SDDL that are Load Balancer, PoA-Manager and Processing Nodes. All nodes showed in Figure 2 will be explained throughout this work.

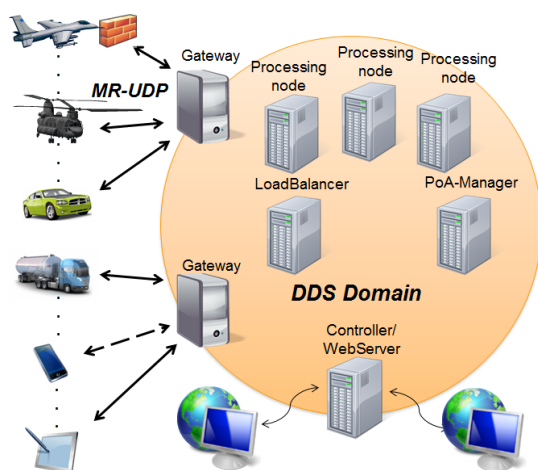


Figure 2. SDDL Architecture

Taking advantage of DDS' distributed P2P architecture and its highly optimized Real-Time Publish Subscribe wired protocol, SDDL is naturally scalable, i.e. new processing nodes or Gateways can be dynamically added to SDDL's core whenever more MNs have to be served, or new data flow processing is required. In regard to the connections with the MNs, whenever some Gateway is overloaded the data flow to and from a large set of MNs, SDDL is capable of seamlessly migrating a fraction of this set of MNs to a underloaded Gateway. This is possible through a SDDL-internal management node, called the PoA-Manager, which continuously monitors the load of each Gateway - in terms of the number of served MNs - and a Client communication library (CNCLib) at the MNs, which accepts both updates of alternative Gateway addresses and/or commands to reconnect to a new Gateway address, from the PoA-Manager. In spite of the unavoidable mobile disconnection, these handovers between Gateways are very fast and completely transparent to the client applications running on the mobile nodes. On the one side, the messages from the MN are buffered in the CNCLib until the new connection is established, and on the other side, messages addressed to the MN are also temporarily intercepted by a SDDL node and then re-routed to the new Gateway, as soon as it signals the connection establishment.

#### B. MAPE-SDDL: an Autonomic Extension for the SDDL

In order to address general dynamic adaptivity requirements for the SDDL middleware, we decided to extend it with autonomic capabilities. This extension, inspired by the MAPE-K loop, is called MAPE-SDDL. The goal is to support resource monitoring, as well as analysis, planning and execution of dynamic reconfigurations on components of the SDDL middleware. It comprises four services: Monitoring Service (MS), Local Event Service (LES), Analysis and Planning Service (APS), and Control and Executing Service (CES).

The MS collects data from any SDDL resources, such as Gateways and Processing Nodes. The monitoring is applied to properties from these resources, such as: CPU load, amount of memory available, network bandwidth and latency, number of served MNs (by each Gateway) or number of DPSs assigned to each Processing Node (see DPS concept in Section IV-A). Each property is associated with a set of operation ranges, which are defined by the framework user. For example, one could use the following operation ranges for monitoring the CPU load usage: [0%,30%], [30%,70%] and [70%,100%]. The MS then notifies the LES (located at the same node) whenever the monitored property switches its operation range, which might indicate a significant change on resource usage.

The LES receives these range change events from the MS and publishes event notifications to subscribed components. Events are occurrences which indicate that a resource

availability condition extended itself throughout a specified amount of time, i.e., its duration time. Event evaluation is based on regular expressions written by application developers or operators, as part of each event definition. For an event notification to be triggered, the corresponding expression must remain valid during the specified duration time. This avoids the generation events when short-lived situations occur (e.g., a CPU load peak on a Processing Node during a few seconds).

The APS analyzes the received event notifications and makes a diagnosis of the problems to be solved. Mobile connection overload on the Gateways, and unbalanced load between Processing Nodes are examples of problems that are already diagnosed by the APS of MAPE-SDDL. After diagnosis, the APS will seek the dynamic reconfiguration actions to resolve the problem, and then build an appropriate action plan. The decision-making for building the plan is defined by the user through the use of rules and a rule processing engine. Each type of reconfiguration action supported by CES receives a UID. This identifier is included in the action plan in order to allow the CES instances to know what reconfiguration actions must be performed. The action plan for mobile connectivity management takes the form of a mandatory handover request to several mobile nodes (with a new Gateway address list) that is generated by the PoA-Manager, an instance of the APS. In the case of the load balancing process, the action plan is a sequence of actions to the Processing Nodes, as will be explained in Section IV-A

Finally, the CES is the adaptation engine that applies the corresponding reconfiguration actions at the resources in response to their availability/load changes. Among the types of dynamic reconfiguration actions supported is the ability of moving DPSs from a Processing Node to another (cf. Section IV-A) The ability to migrate to some set of MNs from one Gateway to another (cf. Section III-A) is also implemented by CES, which in this case, resides in the mobile Client Lib, which performs the disconnection from one Gateway and the reconnection to the new Gateway. In the load balancing process the CES is implemented as part of the Cache manager at each Processing Node.

#### IV. LOAD BALANCING OF MOBILE DATA STREAMS

##### A. Proposed Autonomous Approach

This work proposes a load balancing solution for DDS-based systems named *Data Processing Slice Load Balancing* (DPSLB). The key concept of the proposed solution is the *Data Processing Slice* (DPS), which is the basic unit of load for balancing among server nodes. These nodes will be called *Processing Nodes* (PNs) throughout the text. The general idea is that each PN has some DPS assigned to it, and that load balancing is equivalent to a redistribution of the total number of DPS among the PNs according to their

current load (which is indicated by several metrics, such as CPU and memory utilization).

The types of DDS nodes that compose the DPSLB approach, showed in Figure 2, are *PNs*, which execute the MS, LES and CES (only Execution Service) services, and the *Load Balancer*, which executes the APS and CES (only Control Service) services of the MAPE-SDDL. The Load Balancer is responsible for monitoring the load of PNs, generating the actions to redistribute the system's workload when an unbalance is detected and controlling the actions executed by PNs to move DPSs between them.

As mentioned, the proposed solution relies on the concept of *DPS*, or simply, *Slice*, which represents a percentage of the total system workload being processed by the PNs. Every data item of the data stream (e.g. produced by a mobile node) must have assigned a single DPS, in order to be processed by some PN. If a data item has no associated Slice, it will not be delivered to a PN for processing. Each *Slice* is logical identified by a unique numeric ID (identifier), commonly in a range between zero and the total number of defined *Slices*, minus one. Thus, the DDS Topic carrying application data produced by the publishing nodes has a specific numeric field holding the *Slice-ID* assigned to each data item.

The *Attribution Function*, is responsible for choosing a valid *DPS* for each produced data item. The DPSLB solution requires the Attribution Function to be a very low cost function, since it has to compute/choose a DPS for each produced data item, and this data will probably be produced at a very high rate. This function may be a hash function applied to a field of the data item, to the data producer's ID, or a random value. A good candidate function for this is modulo operator (remainder of division). The Attribution Function does not have to ensure that the data items are uniformly distributed over the total set of *Slices*.

*Load Balancing Process* is the process of moving *Slices* from a PN to another. The process is started when the *Load Balancer* detects a load unbalance of the system and decides that some DPS should be moved to a different PN to reach a better load balance. During this process both *PNs* involved, i.e the Slice-giving and the Slice-taking PN, must work in a coordinated manner so to guarantee that all data items are processed, and only by one of the PNs.

The *Load Balancer* plays the role of coordinator of the actions executed in the *Load Balancing Process*, which are effectively executed by the overloaded and the underloaded PNs. Figure 3 illustrates the redirection of the data stream when *Slice* DPS-5 is moved from PN A to PN B. During the *Load Balancing Process* both PNs receive the data items of DPS-5, but initially none of them will process the data from this DPS. Instead, they store these received data in their local caches. Then, PN A sends its cached items to B. After receiving A's cached items, PN B has to identify the data items that appear in both caches and then generate a *Merged Cache*, which contains all data items of DPS-5 without

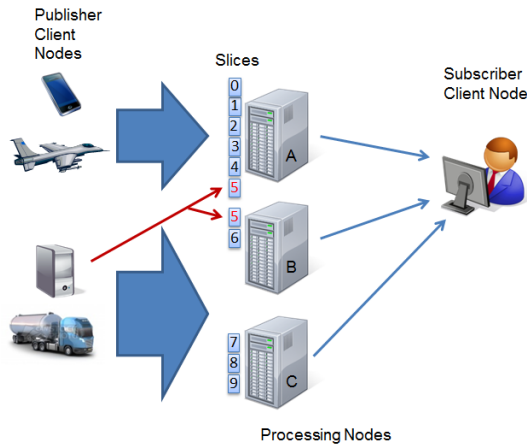


Figure 3. Data flow during Load Balancing Process

duplicates. Thus, the specific sequence of actions sent by the Load Balancer to move a DPS between two PNs are: (i) Update DPS’s state at A to *Cache Data Items*, (ii) Add DPS at B with *Cache Data Items* state, (iii) Remove DPS at A, (iv) Update DPS’s state at B to *Process Data Items* and (v) Send cache from A to B. After this, B will generate and process the *Merged Cache*, and A will continue to process the data of its other Slices. The *Add* and *Remove* actions determine if the corresponding data items are delivered or not, respectively, to a node in a DDS Domain. This is possible by a dynamic adjustment of the subscriber filters.

**B. Preliminary Test Results**

Initial dynamic adaptation tests performed with the MAPE-SDDL middleware have already shown encouraging performance results. Regarding MAPE-SDDL’s connectivity load balancing, we did the following test: We initially connected 600 simulated mobile nodes (MNs) to one Gateway, and then activated a new ‘empty’ Gateway. After a while, the PoA-Manager identified a load unbalance, and requested half of the MNs to migrate simultaneously to the new Gateway. At this bulk handover, all 300 MNs were able to reconnect at the new Gateway in less than 750 ms and none of the data items produced regularly (every 10 seconds) by each of the MNs was lost.

In order to evaluate the DPSLB solution and its implementation, we developed a prototype application (in Java and using CoreDX DDS) that utilizes the DPSLB prototype for balancing of its data processing load. This prototype application consists of clients that publish color images into the DDS domain, and PNs that receive the images, convert them to grayscale and, thereafter inform the corresponding client about completion of the image processing. Both communication paths happen trough two DDS Topics.

Figure 4 illustrates the deployment of the prototype application used for evaluation. Clients publish images through

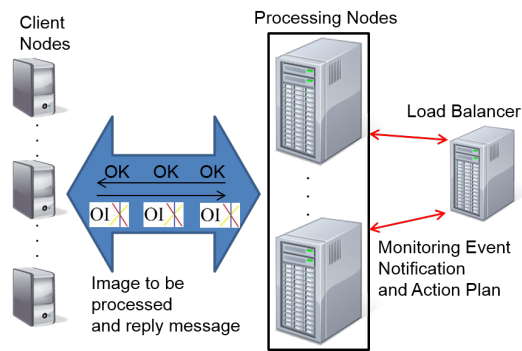


Figure 4. Deployment of the evaluation application

the *ClientTopic* and PN servers reply with completion notifications published into the *ServerTopic*. The *ClientTopic* has fields: *sliceId* (required by DPSLB to produce the merged cache); *id* of the data item; *senderId* to identify the client; *timestamp* to inform the data item creation time and *message*, that carries the serialized image. The *ServerTopic* holds fields: the data item *id*, *timestamp*, *senderId* and *message*, which carries the reply message, a serialized Java String, such as “Processed”. Although this message could as well carry the result image (grayscale), this application prototype sends only a “OK” message, since the content and size of the reply message is irrelevant for evaluating the DPSLB solution. The Load Balancer analyzes the load of PNs and, transparently to the application, balances their image processing workload. It is important to stress that there is no communication, neither directly nor indirectly, between clients and the Load Balancer. Hence, the load generated by clients does not affect the Load Balancer, only the PNs.

DPSLB prototype was tested with data/image publication rates starting from 160 (1,4 MB/s) up to 1.365 (10 MB/s) data items per second. The Attribution Function of choice was the modulo operator applied on the id field, and the number of available slice was chosen to be ten. The setup used for the experiment was the following: 5 PNs, one Load Balancer and a Client simulator deployed on three Physical Machines. executing in a LAN with bandwidth 100 Mbps. Each PN executed on a dedicated Virtual Machine running the Ubuntu [64] 12.04 32-bit Operating System, configured to use one CPU core and 512 MB. The three physical machines had following configurations: Intel i5 4 x 2.66 GHz, 8 GB DDR3 1333 MHz running Windows 7 64-bit; Intel i5 4 x 3.1 GHz, 8 GB DDR3 1333 MHz running Fedora 15 64-bit; and Intel Dual-Core 4 x 2.66, 8 GB DDR2 667 MHz running Mac OS X 10.7.5.

To assess the Load Balancing overhead, we compared the throughput and the mean RoundTripDelays (RTD) of the same image processing application using CoreDX DDS in two configurations: using the DPSLB solution and without

Load Balancing support. The overhead of the DPSLB solution was expressed by percentages (%) of the throughput loss, and the mean RTD increase, respectively. To evaluate the DPSLB overhead, 10,000 data items were produced with a data production rate of 1.150 data items per second (DI/s). The application using DPSLB was able to process 81,044 DI/s and the application without any load balancing support, was able to process 82,194 DI/s. These numbers show an overhead of 1,4% introduced by the DPSLB implementation. Regarding to the RTD, the application using DPSLB had a mean RTD of 60,45 seconds, while the application without DPSLB delivered a mean RTD of 59,51 s. This difference represents an increase of 1,58% on the RTD. The mean time required to complete a Load Balancing Process with a data production rate of 10 MB/s and ten slices was 454 ms.

## V. RELATED WORKS

There is much research and development of autonomic load balancing in middleware for distributed systems, but to the best of our knowledge, there is no other work that leverages the benefits of the MAPE-K model for dynamic adaptiveness in DDS-based systems, and more specifically, proposes a load balancing approach for mobile data stream processing that is reliable, efficient and flexible. However, [9] [10] [11] propose load balancing mechanisms for distributed systems, either for the routing layer or the data processing layer.

The work by Cheung et al. [9] has developed a load balancing mechanism to balance the subscription load among Brokers on the Padres Pub/Sub system, where publishers or subscribers may freely migrate among Brokers. While [9] focuses on the routing layer for a broker-centered Pub/Sub system and clients (publishers or subscribers) are impelled to change their *Brokers* for data flow load balancing, DPSLB solution is based on DDS' P2P architecture for balancing the load among *Processing Nodes*.

REVENGE [10] is a DDS-compliant infrastructure for news dispatching among mobile nodes and which is capable of transparently balancing the data distribution load within the DDS network. In the same way, [10] only load balances the routing substrate, while MAPE-SDDL is able to load balance the mobile connections via PoA-Manager and processing nodes via Load Balancer in the DPSLB.

In [11], a non-coordinated load balancing approach that relies on *Magnetic Fields* is proposed: the idea is that underloaded nodes attract data from overloaded nodes. In an completely opposite way, the Load Balancer in DPSLB performs the MAPE-K tasks of Analysis, Planning and Execution, and carefully synchronizes the re-allocation of Data Processing Slices from one Processing Node to another. This has the advantage of a more efficient and reliable load balancing, but the drawback of the dependability of the Load Balancer.

## VI. DISCUSSION

This paper proposed a novel approach to load balancing mobile connections and data streams based on the MAPE-K model, which entails several advantages that go far beyond a simple boost of performance. Most current load balancing methods are quite inflexible, since they always make same sorts of decision, without considering that the system may require different load distribution approaches depending on the current state of data stream processing (high/low load) or the state of the infra-structure (e.g. node failures, communication failures, re-organization, etc.). Moreover, by being disassociated from any Autonomic architecture, traditional load balancing mechanisms fail to incorporate self-monitoring, self-analysis and self-adaptation behavior as response to changes in the execution environment.

Since our load balancing mechanism is structured according to the MAPE-K model it is able to deliver sustained load balancing performance under various conditions. For example, based on the information collected by MS, CES is capable of adjusting parameters of the load balancing algorithm directly (i.e. parametric adaptation). For other changes of conditions, CES may even substitute the load balancing algorithm by a more effective one. Adaptation can also be used to circumvent failures of Processing Nodes and Gateways. For these cases, the APS can choose the best parameters, algorithms or techniques for handling outage of failed elements, and recovery actions. Finally, it is also possible to implement a machine learning technique in the APS, which would allow the load balancing mechanism to anticipate future change demands, and thus react in a more effective and efficient way. This knowledge about past behavior and adaptation performance of the system would have to be represented and analyzed by the adaptation logic, which is a feature made possible in the MAPE-K model.

Furthermore, our load balancing approach for Data Stream Processing is targeted at DDS-based systems, which support fully decentralized system architectures. It is a generic solution, transparent to the SDDL applications, able to route data streams to Processing Nodes with low overhead and is inherently scalable, i.e. it supports large numbers of nodes and large-volume data streams. The DPSLB Solution works with any type of application object/message and is totally transparent to the application developers, who must only inform which DDS Topics are subject to load balancing by DPSLB. They can still customize their applications with the DDS QoS policies of their choice.

Also, since Processing Nodes can dynamically join or leave the system during operation, DPSLB supports seamless variations of computational resources, i.e. scalable systems.

Finally, the DPSLB also supports customization since several load balancing algorithms (i.e. implemented in APS of MAPE-SDDL) can be applied at the Load Balancer.

## VII. CONCLUSION AND FUTURE WORK

The need for remote monitoring and high performance processing of large mobile data streams in a timely manner is becoming common to many systems such as Intelligent Transportation Systems, Fleet Management and Logistics, and integrated Industrial Process Automation.

The main contribution of this work is the development of a novel approach to load balancing that has two main novelties: its autonomic behavior based on MAPE-K model and the use of DDS as its communication infra-structure. The underlying middleware, MAPE-SDDL, supports not only load balancing of mobile connections among different Gateways but also node balancing of data stream processing across multiple Processing Nodes. To the best of our knowledge MAPE-SDDL is the first middleware that has developed an autonomous load balancing approach tailored to DDS-based systems. Preliminary performance evaluations have shown encouraging results what motivate us to continue the development of SDDL and its autonomic extensions.

By being disassociated from any autonomic reference model, traditional load balancing mechanisms fail to incorporate self-\* properties, which are the pillars for the development of more adaptive and scalable systems. Moreover, most of the traditional load balancing approaches are not well suited for high-throughput mobile communication and data stream processing systems, as they are not based on a communication layer with real-time communication capabilities. On the other hand, our load balancing approach was specially designed for decentralized systems based on the DDS standard, and hence is capable of fulfilling application requirements such as real-time and high throughput data communication and processing, scalability and fault tolerance.

Preliminary tests have yielded encouraging performance result, which motivate us to proceed with the development of SDDL's adaptivity and load balancing capabilities. For a data stream production rate of 10 MB/s, DPSLB is able to complete the Load Balancing Process of 5/10 DPS in 454 ms. And regarding load balancing of mobile node (MN) connections, MAPE-SDDL is able to migrate 300/600 MNs from one Gateway to another Gateway in less than 750 ms.

As future work, we are planning the design, implementation and evaluation of other load balancing algorithms, both for data stream processing and connectivity load distribution, as well as developing support for general state transfers among the managed resources (Processing Nodes or Gateways) during Load Balancing Process. We also plan to design and implement a new component in MAPE-SDDL, called Distributed Event Service (DES), which will allow the detection of composite events made of basic events from different event sources (e.g., distributed Processing Nodes).

## ACKNOWLEDGMENT

“This work is partly supported by project Mobile InfoPAE and CNPq scholarship n°. 310253/2011-0”

## REFERENCES

- [1] S. Karnouskos and A. Colombo, “Architecting the next generation of service-based scada/dcs system of systems,” in *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, November 2011, pp. 359–364.
- [2] G. Pardo-Castellote, “OMG Data Distribution Service: Architectural Overview,” in *Proc. of the IEEE Military Communications Conference (MILCOM '03)*, vol. 1, October 2003, pp. 242–247.
- [3] M. Xiong, J. Parsons, J. Edmondson, H. Nguyen, and D. C. Schmidt, “Evaluating the Performance of Publish/Subscribe Platforms for Information Management in Distributed Real-time and Embedded Systems,” 2010.
- [4] L. D. Silva, R. Vasconcelos, R. A. Lucas Alves, G. Baptista, and M. Endler, “A communication middleware for scalable real-time mobile collaboration,” in *Proc. of the IEEE 21st International WETICE, Track on Adaptive and Reconfigurable Service-oriented and component-based Applications and Architectures (AROSA)*, June 2012, pp. 54–59.
- [5] I. Twin Oaks Computing, “Twin oaks computing, inc.” April 2012. [Online]. Available: <http://www.twinoakscomputing.com/>
- [6] J. O. Kephart and D. M. Chess, “The vision of autonomic computing,” *Computer*, vol. 36, pp. 41–50, January 2003. [Online]. Available: <http://dx.doi.org/10.1109/MC.2003.1160055>
- [7] IBM, “An architectural blueprint for autonomic computing,” *IBM White Paper*, 2006. [Online]. Available: <http://www-03.ibm.com/autonomic/pdfs/ACBlueprintWhitePaperV7.pdf>
- [8] “Scalable data distribution layer - overview, use instructions and download,” 2012. [Online]. Available: <http://www.lac-rio.com/sddl/>
- [9] A. K. Y. Cheung and H.-A. Jacobsen, “Load Balancing Content-Based Publish/Subscribe Systems,” *ACM Transactions on Computer Systems*, vol. 28, no. 4, pp. 1–55, December 2010. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1880020http://portal.acm.org/citation.cfm?doid=1880018.1880020>
- [10] A. Corradi, L. Foschini, and L. Nardelli, “A DDS-compliant infrastructure for fault-tolerant and scalable data dissemination,” in *The IEEE symposium on Computers and Communications*. IEEE, June 2010, pp. 489–495. [Online]. Available: [http://ieeexplore.ieee.org/xpls/abs\\_all.jsp?arnumber=5546756http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5546756](http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5546756http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5546756)
- [11] A. Calsavara and L. A. P. Lima Jr., “Scalability of Distributed Dynamic Load Balancing Mechanisms,” in *ICN 2011 The Tenth International Conference on Networks*, no. C, 2011, pp. 347–352.

## QoS-Aware Component for Cloud Computing

Inès Ayadi, Nöemie Simoni

Engineering School Telecom ParisTech  
Paris, France

(ines.ayadi; noemie.simoni)@telecom-paristech.fr

Gladys Diaz

L2TI, Université Paris 13, Sorbonne Paris Cité  
Villetaneuse, France

gladys.diaz@univ-paris13.fr

**Abstract**—Providing dedicated cloud services that ensure user's QoS requirements is a big challenge in cloud computing. Currently, cloud services are provisioned according to resources availability without ensuring the expected performances. The cloud provider should happen to evolve its ecosystem in order to meet QoS-awareness requirements of each cloud component. In this paper, we consider two important points which reflect the complexity introduced by the Cloud management: QoS-aware and self-management aspects. QoS-aware aspect involves the capacity of a service to be aware of its behavior. Self-management implies the fact that the service is able to react itself with its environment. In this paper we propose to integrate a QoS agent in each cloud component in order to control and inform the system about its current behavior.

**Keywords**—Cloud computing; QoS-aware; Autonomic components; self management; Fractal ADL

### I. INTRODUCTION

Cloud computing is a new trend that enables resources (Infrastructure, Platform or Software) to be exposed as services. These resources are offered using a pay-as-you-use pricing plan. The final service offered to the user consists in a set of components, which may be offered by different providers. To satisfy the request of customer, the final service must be provided in accordance with the required level of QoS (Quality of Service). QoS management must be considered to provide the attended end-to-end (E2E) QoS level. In current solutions, a degradation of a component can produce the degradation of the global service. Thus, one of the major challenges in the current cloud solutions is to provide the required services according to the QoS level expected by the user.

Cloud users expect the system to guarantee the required QoS services regardless of any unforeseeable events. Consumers needs vary unpredictably depending on their types (developer, service provider, end user) and their strategies (QoS requirement, cost effective, optimization, etc.). These unstable demands can result in SLA (Service Level Agreement) violation due to the QoS degradation of the cloud services. While cloud providers can ensure the elasticity, the high availability and the reliability of services, the QoS expectations of users are not achieved. QoS is a very important aspect that must be considered in the different phases of life-cycle of Cloud solutions. Thus, the QoS aspects should be considered from the design phase of cloud components in order to achieve the expected QoS requirements. Consequently, we think that QoS-aware is a good approach to be implemented in future dynamic cloud

environments. The QoS-aware approach implies the knowledge of significant QoS information related to each life-cycle phase. In fact, Cloud environments require a dynamic configuration and management of services and resources at run time in order to ensure the expected QoS. This dynamicity and adaptability is only possible if the system is able to use the pertinent QoS information in order to predict the suitable consummation of resources needed by the applications.

Complementarily, self-management approach must be introduced to guarantee the E2E behavior of the global service. The self-management implies the ability of each service component to manage itself its behavior.

We propose in this paper the modeling of both aspects: the QoS-awareness and the autonomic management in the cloud. We suggest the modeling a QoS self-managed cloud component using the Fractal ADL. Our work is part of the OpenCloudware project [2].

This paper is organized as follows. Motivations are presented in Section II. The related work for QoS-aware and autonomic components aspects in cloud is described in Section III. Section IV gives a brief review about our QoS generic model. Our propositions for a QoS-aware component in cloud are presented in Section V. We give the modeling description of our proposed component and a use case in Section VI. Finally, in conclusion, we exhibit the advantages of our approach in Cloud Computing and the future works.

### II. MOTIVATIONS

The present work is a continuation of the previous research studies performed by the UBIS project [5]. This project focuses on the user-centric approach and it aims at providing personalized services anytime, anywhere and anyhow. The goal of this project has been to ensure E2E QoS requirements of the services demanded by the end-user. For this purpose, the QoS management is handled within different layers: service, network and equipment. The resources in different layers are conceived as a set of service elements (building blocks) in order to have a fine grained QoS description. A generic QoS model [1] is proposed for define the service elements behavior. We suggest that this model could be used to deal with cloud computing issues that are related to self-management and QoS-awareness. Consequently, we apply our generic QoS model to design cloud components by considering the expected QoS requirements in the whole life-cycle. In fact, in our point of view, the cloud environment is described as a set of distributed components. A Cloud component is an independent element that provides a well known



functionality and a QoS level. This notion of component is generic and can be applied at different cloud levels. An IaaS (Infrastructure as a Service) component can be for example the CPU (MIPS), the VM, the network switch, etc. In the PaaS (Platform as a Service) level, a component can be an application container, a routing stack (number of packet sent, delay of treatment of each packet), etc. The SaaS (Software as a Service) component is the final service (processing time, requests number, etc.) such as a financial service, a Web service, etc. We also consider that management, control, monitoring and security functions are conceived as cloud components. This fine-grained modeling of components allows composing applications by considering E2E QoS requirements.

To more explain this, consider a user that demands a 3-tier application from the cloud provider. This application is composed by an Apache Web server, a Jonas application server and a MySQL database server. By applying our model in Jonas for example, we consider this tier as a composition of several Cloud components including: CPU, memory, VM (where the Jonas server is hosted), the software of the server, the network binding, etc. Each component is described by its behavior. This allows the deployment of the Jonas software in the adequate virtual environment that is in turn be hosted in a suitable physical environment. In order to satisfy the E2E QoS requirements (SLA), the cloud provider should deploy the 3-tier application by composing adequate cloud components. For example, the Jonas server should be put up in a large VM instance (in terms of CPU and memory) in order to process requests in less than 0.1s. Consequently, the self-acknowledgement of the component's behavior allows the mapping of each virtual application (Vapp) in the suitable environment.

Furthermore, our model deals with the unified control of QoS aspects, and provides an answer to guarantee the required QoS services regardless of any unforeseeable events. *To do this, a QoS-agent will be integrated into cloud components in order to perform dynamic adaptation according to QoS requirements.*

### III. RELATED WORK

Providing a QoS-aware solution requires autonomic capabilities in the cloud components. In this section, we review some approaches treating the two aspects: QoS-aware and self-management components in cloud computing.

#### A. QoS-Aware cloud

S. Ferretti et al. [6] proposes a QoS-aware cloud architecture that aims to satisfy QoS requirements of the application. The principal function of this architecture is the efficient resources management of the virtual execution environment associated to the application. This architecture includes features that eliminate resources over-provisioning by changing and configuring the amount of resources dynamically. But how can describe the behavior of an application?

R. Nathuji et al. [7] propose "Q-Cloud", a QoS-aware control framework that manages resources allocation in order to alleviate consolidated workload interference problem. The

principal aim of this framework is to dynamically allocate resources of co-hosting applications based on QoS requirements. Q-States (QoS states) notion is proposed in order to assign additional QoS levels to the application. The goal of these states is to offer additional flexibility to the user in order to easily improve his application-specific QoS level. But QoS levels generate different behaviors, then, can we talk about the same service?

That is why, QoS monitoring feature presented in [9] according to "as a service" paradigm is interesting. This facility ensures a continuous control of QoS attributes in order to avoid SLA violation.

H. Nguyen Van et al. [8] attempt to manage autonomic virtual resources for hosting cloud services. It proposes a two-level architecture that separates application's QoS specifications from the allocation and provision of resources. An application-specific local decision module is proposed within each application in order to analyze the QoS requirements of the hosted service. This module determines a high-level performance goal in order to make the best decision in allocation and provision phases.

While the cited solutions aim to satisfy QoS requirements of applications, the management is still resource-based by adapting resource reservation to QoS requirements. The QoS-aware aspects are not addressed. A QoS-aware component should provide the same expected service and the same intended QoS level. In others words, Cloud Services must maintain the same behavior even if the environment conditions are changed. Thus, we propose, through this paper, a QoS-aware Cloud component that can itself control its behavior.

#### B. Self-managed components in cloud

An autonomic cloud components should intrinsically integrates the dynamic adaptation and self-management capabilities in order to meet the non-functional requirements. We present in this section a review of some references of this context.

F. Zambonelli et al. [10] propose the management of service components that are able to adapt dynamically their behavior according to the changes perceived in their environment. The research issues include the identification mechanisms, to enable components to self-express the most suitable adaptation scheme and acquiring the proper degree of self-awareness to enable putting in action self-adaptation and self-expression schemes. The rule execution model provides mechanisms to dynamically detect and handle rule conflicts for both, behavior and interaction rules.

H. Liu [11] et al. propose an "Accord framework" that enables the development of autonomic elements and their autonomic composition. They provide rules and mechanisms for reconciliation among manager instances, which is required to ensure consistent adaptations. For example, in parallel Single Component Multiple Data (SCMD) each processing node may independently propose different and possible conflicting adaptation behaviors based on its local state and execution context. Several others research papers have been related on the self-management of cloud services such as [12], [13], and [14]. They implement the loop MAPE

principles (Monitoring, Analysis, Planning, and Execution) in order to maintain QoS requirements and reduce the probability of SLA violations.

These approaches suggest some paradigms to create autonomic cloud components, but they still modeled according to the monolithic approach. However, cloud computing system requires a new approach where Cloud Services are conceived as a set of independent building blocks. Moreover, these proposals focus on self-management treated by the implementation of loop MAPE principles. But, in an environment as heterogeneous as the cloud, these solutions can be difficult to implement. Is that self-management would not be a more appropriate solution?

In our approach, we consider the self-management of each service component based on QoS control. The QoS control helps to inform and prevent the degradation of component's behavior. It provides a new approach to preserve and guarantee the services of the whole System. To do this, we propose to integrate a QoS agent into the cloud component in order to allow it to manage his behavior. In fact, the integration of a QoS agent in each component allows implementing the self-management capabilities, since the agent is able to indicate whether or not the component performs its work in the normal conditions. For example, if the current values (at runtime) of a QoS criterion is exceeded over its thresholds values, the QoS-agent sends an "out-contract" to indicate that there is a problem and that it must be replaced by another ubiquitous component. To complete the self-management mechanism, we assume the existence of ubiquitous components in the Cloud environment. These ubiquitous components provide the same service and have the same QoS level.

IV. QoS GENERIC MODEL OVERVIEW

The principal objective of our generic QoS model is to design components by taking into account not only the processing aspects but also the management ones. For this purpose, an informational model is proposed to manage the non-functional aspects of each resource (see Fig. 1). This model is generic and unified. It applies to all layers (equipment, network, service). Resources in each layer are conceived as a set of service elements. The informational model describes the QoS criteria of each service element.

Our QoS model introduces the definition of autonomic components throughout a QoS agent. The QoS agent manages and monitors the component behavior by using QoS criteria types (see Table I).

TABLE I. QoS CRITERIA TYPES

		QoS Criteria			
		Availability	Capacity	Delay	Reliability
Description	is the portion of time that a service component makes the requested service without failure	is the processing capacity of service component during a unit of time	is the total time taken by a service component to fulfill its functions	is the compliance rate of the rendered service compared to the demanded one	

Resources of each component must be allocated dynamically in accordance with their current QoS. The management of the sharing service's resources and capabilities is performed through a queue integrated in each component. The acceptance of a new request in the queue is done according to the current values of QoS criteria [1]. The QoS criteria are evaluated through three types of measurable values: conception, current and thresholds values [3] (see Table II).

TABLE II. QoS CRITERIA VALUES

		Value types of QoS criteria		
		Conception	Threshold	Current
Description	is the maximum capacity of the service component processing	is the limits values not to be exceeded by a service component in order to ensure a normal behavior	is the real current value of a QoS criteria in instant t. It is used to supervise the behaviour of the service component. This value would be compared with the threshold values to control the non-violation of the service capacities	

The Fig. 1 shows our information QoS model that defines all these concepts. The first level represents the QoS criteria (availability, reliability, delay, and capacity) of each component. The second one shows the measured values of each criterion. Finally, the third level depicts necessary parameters to do measurement.

Our QoS model is based on four criteria: Availability, Delay, Capacity and Reliability. We briefly justify the choice of the four criteria. Our logic is how evaluate the behaviour of a given environment without being linked to its dependencies such as location, time, network type, the delivered service, the terminal, etc. Therefore, our objective is to determine QoS criteria that achieve the End-to-End transparency. We expose the transparency in four dimensions:

- Temporal transparency: a given information can be delivered anytime. This transparency dimension is associated with the availability criteria in order to evaluate how long the system (middleware, network, etc.) is in operation during the transfer.
- Distance transparency: a given information can be delivered regardless of the distance between the end-nodes. A delay criterion is associated to this dimension in order to evaluate the processing and transfer time.
- Spatial transparency: a given information can be delivered regardless of its volume. The Capacity criterion is associated to this dimension in order to evaluate the system capabilities to treat any volume of information.
- Semantic transparency: a given information can be delivered without alteration of its content. The reliability criterion is associated to this dimension in order to evaluate how the system can treat correctly the information.

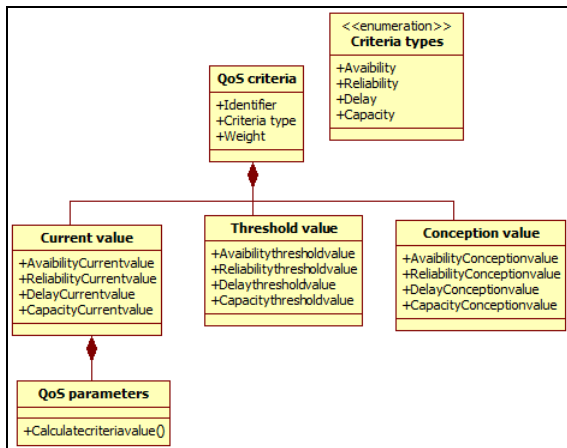


Figure 1. Information QoS model

Our QoS generic model is associated to multiple profiles [4] for managing QoS measured parameters, which are solicited during the different phases (design, deployment, operational) of the life-cycle (see Table III).

TABLE III. PROFILES IN LIFE-CYCLE

	Life-cycle phases		
	Design	Deployment	Operational
Profiles	Resource profile	Resource Usage Profile	Real Time Profile
Values of QoS criteria	Conception values	Threshold values	Current values

### V. QoS-AWARE COMPONENT PROPOSITION

Our contribution consists of adding a new feature to the Cloud Service Component (CSC) in order to cope with the QoS management throughout the life-cycle. Our extension of Fractal component is an integration of a QoS component that represents the QoS agent presented in Sections II and IV. This "QoSComponent" allows managing the cloud component's behavior and enables to send notifications in the case of SLA violation or QoS degradation. We describe in this section our proposed QoS-Aware component.

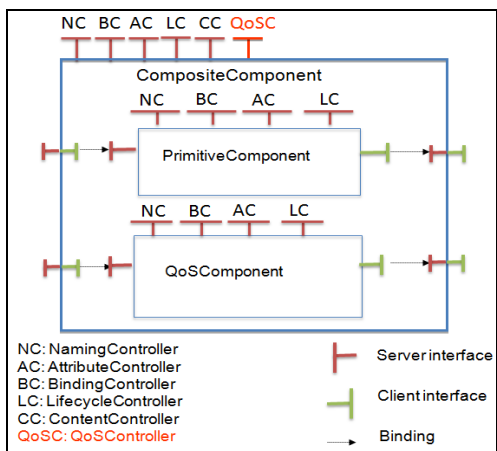


Figure 2. QoS-aware Cloud Service Component with Fractal

#### A. QoS-aware Component Model Description

Our proposition consists in an extension of the Fractal component by adding non-functional features that manage dynamically the offered QoS, like is shown in Fig. 2. The goal is to add a QoSComponent within each Fractal component. A new Fractal control interface (QoSC) is proposed to monitor and manage the QoS criteria.

#### B. Characteristics of of the CSC

The CSC are characterized by the following proprieties:

- Mutualisation: the CSC is a multi-tenant service element. Several users can share it in the same time. A CSC is stateless in order to offer the same service to all simultaneous demands.
- QoS Self-management: a CSC can self-control and self-monitor his behavior. The QoSComponent is implemented in each CSC in order to monitor the QoS criteria and to generate accurate notifications in the case of QoS degradation.
- Exposability: a CSC has a business value. Users can custom their services thanks to a portal catalogue.

#### C. Functionnals aspect of the CSC

The functional aspects represent the implementation of the offered service, including the content and the management controllers. The proposed CSC uses the following native Fractal controllers:

- Attribute Controller (AC): it manages (get, set, and update) the configuration attributes of the component. In the reconfiguration phase, this controller can modify the values of these attributes.
- Binding Controller (BC): it manages interconnections between client and server interfaces. Binding channels can be deactivated or activated depending on Fractal component life-cycle.
- Content Controller (CC): this controller manages the hierarchic architecture of the Fractal component. It adds, removes or substitutes Fractal sub-components.
- Life-cycle controller (LC): it allows the start-up and the shutdown of a Fractal component. As the QoS management is performed at many phases of the life-cycle, we propose new functionalities to this controller that will be described in the section IV.4.
- Naming controller (NC): it manages the Fractal component identification.

To ensure the QoS self-management of the cloud component, we propose a new QoS Controller (QoSC). The QoSC controller manages the CSC's behavior. This controller allows sending QoS notifications that indicate if the component maintains its behavior. These notifications are essential to take reactive decisions in the case of failures or QoS degradation.

#### D. The QoS management in the Life-cycle

The native life-cycle controller of a Fractal component allows managing the runtime phase. It handles the start-up and the shutdown states. However, the autonomic

management of the component's behavior requires more operations in others life-cycle phases. In fact, the QoS management is not only performed in runtime phase but also in the design and the deployment phases.

The life-cycle controller is needed to check which is the current life-cycle stage of the component and what are the constraints associated with each phase. For this purpose, we propose new functionalities of the life-cycle controller in order to maintain QoS measurements according to life-cycle phases. We define six life-cycle phases of a cloud component: design, development, deployment, runtime, billing and retirement. In the present subsection, we focus on life-cycle phases in which QoS management is performed (design, the deployment and the runtime phases).

- Design phase: represents the modeling phase of a cloud component. In this phase, each QoS criteria has conception values that determine the maximal cloud service processing capacities (e.g., server memory, CPU cores, maximal transaction per second, etc.). These values are static and unchangeable during the whole life-cycle. The life-cycle controller creates the “resource profile” containing these values.
- Deployment phase: represents the integration stage of a cloud service within the execution environment. In this phase, the life-cycle controller determines constraints of the execution environment and creates the “resource usage profile” with the threshold values of each criterion. These values show the limited capacity beyond which the cloud service's behavior becomes abnormal (e.g., the limit CPU of the virtual machine in which the cloud service is deployed).
- Runtime phase: represents the processing phase of a cloud component. The “current values” of each QoS criteria are dynamically determined throughout this stage (e.g., free disk space, current load network, requests number in the queue, etc.). These values are measured and updated by the QoSComponent and containing in the “Real-time profile”.

In this stage, the life-cycle controller manages the cloud services states that are presented as follows:

- Unavailable: this state is equivalent to the shut-down state performed by the native life-cycle controller.
- Available: this state indicates that the CSC is not reserved and it is able to be used.
- Activable: in this state, the CSC is awaiting for additional information (example: login/password) to begin the execution.
- Activated: is equivalent to the start-up state. In this phase, the CSC is already in use.

E. QoS Component description

In our proposition, a QoS component is integrated in each CSC in order to allow the QoS self-management. The QoSComponent controls the CSC's behavior throughout the QoS interface. The QoS component has two main functions: cloud service control and QoS notifications.

a. Cloud service control

The QoS component controls the behavior of a cloud service element based on QoS profiles (Section III.B). In fact, each cloud service possesses a set of QoS profiles that include criteria QoS values (conception, current and threshold). The QoSComponent performs the following functions by means of the QoSC controller.

- Update current values of the QoS criteria: the QoS component interrogates a "Monitoring module" in order to have current values of the QoS metrics. Then, it evaluates and updates these values in the "real time profile". The Monitoring module is an entity that gives different metrics others than QoS metrics. This module is provided as a service (MaaS: Monitoring as a Service) and integrated in each cloud component. The description of this module is out of the scope of the present paper.
- QoS degradation: the QoS component has to detect degradation behavior of the cloud service. In fact, there are many causes that can bring to QoS degradation such as network congestion, increasing processing time, etc.
- Processing a new request: as the cloud service element can be shared by several users in the same time, new users' requests can be received. This component uses the QoS profiles in order to make an accurate decision about the possibility to treat a new request.

b. QoS notification (In/Out Contract)

The QoSComponent notifies permanently if the service retains his behavior during the run-time. It sends to the Cloud system an "In contract" message if the intended comportment is maintained (Figure 3). The second possible notification is an "Out contract" message. This notification indicates that the CSC does not maintain the correct behavior. This notification helps to prevent the occurrence of anomalies or failures.

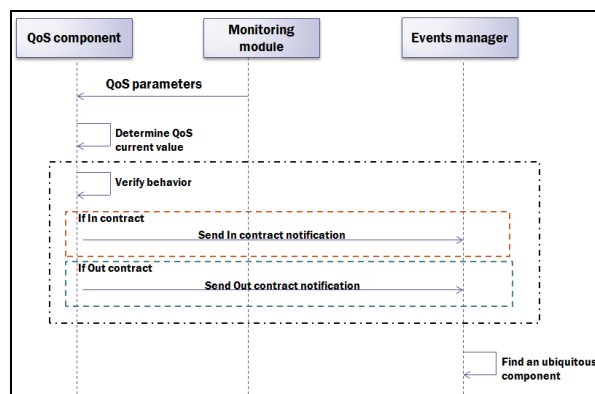


Figure 3. QoSComponent notifications

If the events manager receives an Out contract. One solution may be performed is to replace the degraded CSC component by a ubiquitous one. In fact, two CSC components are ubiquitous if they have the same function

and an equivalent QoS level. Many others reactions can be performed by the events manager. However, the management reactions will be treated in our future work.

## VI. SPECIFICATION AND USE CASE

In this Section, we describe the CSC specification with the Document Type Definition (DTD) grammar.

### A. Specification of the extended DTD

To describe distributed cloud component in accordance with our models we need to specify new formal notions with a DTD grammar. To do this, we use the basic DTD grammar of the ADL fractal language. We propose an extension of this grammar in order to depict the proposed notions such as QoS component, QoS controller, etc. All implementations presented in this paper have been made by using Fractal ADL plug-in [15] in the eclipse IDE (Integrated Development Environment) [16]. This Section shows the extended DTD grammar that describes the notions corresponding to our model.

The basic DTD notions that define the native Fractal component specification are the following:

- component: can be primitive or composite
- interface: is the point of access to the component
- binding: it allows components communication
- content: represents the content of the component
- attributes: are described by a name/value pair. They are used to (re)configure the component
- controller: represents the membrane of a component.

As it is shown in Fig. 2, an Autonomic Cloud Service component is essentially composed of three parts: the interfaces, the primitive fractal component and the QoS component. Based on the basic DTD description, we add new notions to describe our proposed component. These notions are the following:

- CompositeComponent: represents our proposed component (the Cloud Service Component).
- PrimitiveComponent: represents a non-composite Fractal component.
- QoSComponent: manages the non-functional aspects of the cloud component (Section IV).
- interface-QoSC: is the QoS notifications controller.

The remainder of this Section describes examples of the extended DTD grammar. We describe the Cloud Service Component with the extended DTD grammar as follows:

```
<ELEMENT CompositeComponent (PrimitiveComponent+,
QoSComponent+, NC+, BC+, AC+, LC+, CC+, interface-QoSC+)>
<ATTLIST CompositeComponent
name CDATA #REQUIRED >
```

The DTD of the QoSComponent is described through three notions: QoSCriteria and QoSParameter. The needed controllers of this element are: NC, BC, AC, LC. A QoSComponent has a name and a role (client, server).

```
<ELEMENT QoSComponent (QoSCriteria+, QoSParameter, NC+, BC+,
AC+, LC+)>
<ATTLIST QoSComponent
```

```
name CDATA #REQUIRED
role (client | server)>
```

As previously mentioned, the interface-QoSC is a new controller added to the CompositeComponent. Through this interface the QoS component communicates with the component to send management messages: In/Out contract. The QoS component can have different roles (passive, active, proactive and inter-active).

```
<ELEMENT interface-QoSC (#PCDATA)>
<ATTLIST interface-QoSC
name CDATA #REQUIRED
role (passive | active | proactive | inter-active)
signature CDATA #REQUIRED >
```

Four QoS criteria (Availability, Delay, Capacity, Reliability) define the type of parameters to be measured. Our model defines three values types of these criteria: (Conception | Threshold | Current). The DTD specification of the QoSCriteria is as follows:

```
<ELEMENT QoSCriteria (#PCDATA)>
<ATTLIST QoSCriteria
Criteriatype (Availability | Delay | Capacity | Reliability)
ValueType CDATA #REQUIRED
roleValueType (Conception | Threshold | Current)>
```

### B. Use case description

In order to show a proof of concept of our propositions, we describe the example indicated in Section II. The goal is to apply our model to describe a simple distributed application in the context of cloud computing. The use case in Fig.4 represents a requested PaaS service defined by three software components: Apache, Jonas and MySQL. Each component is installed in a different VM (ApacheVM, JonasVM and MySQLVM). These VMs are interconnected throughout two links: AJP and JDBC (Figure 4).

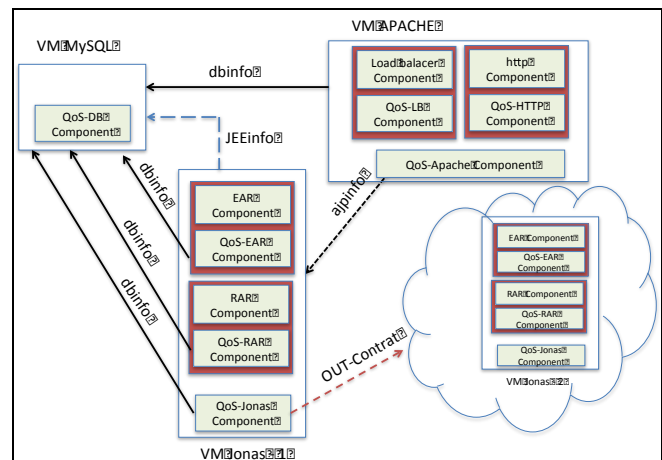


Figure 4. Use Case: Springoo

These components are in their turn composed of others sub-components. The Apache Server is composed of two sub-components: Load Balanced and HTTP Server. The JEE applications are deployed in each Jonas component. They are

composed by EAR and RAR components. According to our model each component is described by its functional and non-functional (QoS-component) aspects. For example each EAR component is described by the EAR component (functional part) and the associated QoS-component (non-functional part). The QoS component manages the behavior of each component by controlling the QoS criteria. We also suppose two ubiquitous Jonas components located in different cloud environments (VM Jonas-1 and VM Jones-2).

We propose the following two scenarios to show the QoS-awareness and self-management of the Jonas component:

- The Jonas component receives more requests than it is able to treat (peak load). In this case, the QoS component notifies that the current capacity is exceeded and sends an Out contract notification. Then, the Jonas component rejects the request.
- If the Jonas component is not available, the QoS interface sends an Out contract. To deal with this, a ubiquitous Jonas component (VM Jones-2) is demanded to replace the failed one. We do not describe in this paper the mechanism how to choose the new component, it is out of the scope of this paper.

## VII. CONCLUSION AND FUTURE WORK

Cloud computing is a new paradigm that provides on-demand services over the Internet. Cloud services are viewed as a composition of distributed components. These components have multiple types: infrastructure (hardware, storage, network), platform or software. On-demand, flexibility and availability of computing components are behind the great success of cloud computing. However, the QoS requirements of a cloud user are still not guaranteed.

To ensure the QoS requirements, the cloud services must be able to adapt its behavior dynamically. In this paper, we considered two important points to reflect the complexity introduced by the QoS cloud management: the self-management and the QoS awareness. We present the mapping of our generic QoS model to deal with these two aspects. Two principal propositions are presented:

- an integration of our QoS model to conceive a new QoS-aware cloud component (CSC).
- an extension of a DTD basic grammar in order to describe our QoS model specification. This DTD is used to describe the CSC component through the Fractal ADL language.

We are based on our generic QoS model to propose an autonomic cloud component that is able to manage its non-functional aspects. We use our informational model to maintain the QoS self-management aspects. We present a QoS component, which is integrated in each cloud component. This component uses the QoS criteria to control the current behavior of the CSC component and to inform the system about the current component's state ("IN/OUT contract").

The present work represents the first step to study how the self-management of a QoS-aware cloud component behavior is achieved. In the further work, we will present

some mechanisms to maintain the autonomic loop principles in our proposed component.

## ACKNOWLEDGMENT

This work is supported by the OpenCloudware project. OpenCloudware is funded by the French FSN (Fonds national pour la Société Numérique), and is supported by Pôles Minalogic, Systematic and SCS.

## REFERENCES

- [1] ETSI TR 102 805-3 V1.1.1 (2010-04). Part 3: QoS informational structure. On-line report: [http://www.etsi.org/deliver/etsi\\_tr/102800\\_102899/10280503/01.01.01\\_60/tr\\_10280503v010101p.pdf](http://www.etsi.org/deliver/etsi_tr/102800_102899/10280503/01.01.01_60/tr_10280503v010101p.pdf) [retrieved: February, 2013]
- [2] <http://www.opencloudware.org/bin/view/About/ProjectInfo> [retrieved: January, 2013].
- [3] ETSI TR 102 805-1 V1.1.1, Part 1: User's E2E QoS – Analysis of the NGN," On-line report : [http://www.etsi.org/deliver/etsi\\_tr/102800\\_102899/10280501/01.01.01\\_60/tr\\_10280501v010101p.pdf](http://www.etsi.org/deliver/etsi_tr/102800_102899/10280501/01.01.01_60/tr_10280501v010101p.pdf), [retrieved: February, 2013].
- [4] N. Simoni, C. Yin, and G. Du Chén, "An intelligent user centric middleware for NGN: Infosphere and ambient grid," in Proc. Communication Systems Software and Middleware and Workshops, COMSWARE 2008, pp. 599-606.
- [5] G. Diaz, K. Chen, N. Simoni, and N. Ornelas, "Spécifications des composants fonctionnelles de la session UBIS," May 2010, on-line report: <http://www-l2ti.univ-paris13.fr/~ubis/UBIS-SITE/FichiersPDF/SP31.pdf> [retrieved: February, 2013].
- [6] S. Ferretti, V. Ghini, F. Panzieri, M. Pellegrini, and E. Turrini, "Qos-aware clouds," in Proc. Cloud Computing, CLOUD 2010, pp. 321–328.
- [7] R. Nathuji, A. Kansal, and A. Ghaffarkhah, "Q-clouds: managing performance interference effects for qos-aware clouds," in Proc. European Conference on Computer Systems, EUROSYS 2010, pp. 237–250.
- [8] H. Nguyen Van, F. Dang Tran, and J.M Menaud, "Autonomic virtual resource management for service hosting platforms," in Proc. Workshop on Software Engineering Challenges of Cloud Computing, ICSE 2009, pp. 1–8.
- [9] L. Romano, D. De Mari, Z. Jerzak, and C. Fetzer, "A novel approach to qos monitoring in the cloud," in Proc. Data Compression Communications and Processing, CCP 2011, pp. 45-51.
- [10] F.Zambonelli, N.Bicocchi, G.Cabri, L.Leonardi, and M.Puvianil, "On Self-adaptation, Self-expression, and Self-awareness in Autonomic Service Component Ensembles," in Proc. Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2011, pp. 108-113.
- [11] H.Liu, M.Parashar, and S. Hariri "A component based programming model for autonomic applications," in Proc. International Conference on Autonomic Computing, ICAC 2004, pp. 10-17.
- [12] I. Brandic, "Towards self-manageable cloud services," in Proc. Computer Software and Applications Conference, COMPSAC 2009, Vol 2, pp. 128–133.
- [13] M. Maurer, I. Brandic, V. Emeakaroha, and S. Dustdar, "Towards knowledge management in self-adaptable clouds," in Proc. SERVICES 2010, pp. 527–534.
- [14] R. Ranjan and R. Buyya, Special section on Autonomic cloud computing: technologies, services, and applications, in Proc. Concurrency and Computation: Practice and Experience, 2012, Vol 24, pp. 935-1034.
- [15] <http://fractal.ow2.org/f4e/>, [retrieved: January, 2013].
- [16] <http://onjava.com/pub/a/onjava/2002/12/11/eclipse.htm>, [retrieved: January, 2013].

# Autonomous Systems: from Requirements to Modeling and Implementation

Nikola Šerbedžija

Fraunhofer FOKUS

Berlin, Germany

nikola.serbedzija@fokus.fraunhofer.de

**Abstract**—Developing autonomous systems requires adaptable and context aware techniques. The approach described here decomposes a complex system into service components – functionally simple building blocks enriched with local knowledge attributes. The internal components’ knowledge is used to dynamically construct ensembles of service components. Thus, ensembles capture collective behavior by grouping service components in many-to-many manner, according to their communication and operational/functional requirements. Linguistic constructs and software tools have been developed to support modeling, validation, development and deployment of autonomous systems. A strong pragmatic orientation of the approach is illustrated by two different scenarios.

**Keywords**—autonomous systems; component-based system; context-aware systems

## I. INTRODUCTION

Developing massively distributed systems has always been a grand challenge in software engineering [1,2,3]. Incremental technology advances have continuously been followed by more and more requirements as distributed applications grew mature. Nowadays, one expects a massive number of nodes with highly autonomic behaviour still having harmonized global utilization of the overall system. Our everyday life is dependent on new technology which poses extra requirements to already complex systems: we need reliable systems whose properties can be guaranteed; we expect systems to adapt to changing demands over a long operational time and to optimize their energy consumption [4,5].

One engineering response to these challenges is to structure software intensive systems in ensembles featuring autonomous and self-aware behaviour [6,7]. The major objective of the approach is to provide formalisms, linguistic constructs and programming tools featuring autonomous and adaptive behavior based on awareness. Furthermore, making technical systems aware of the energy consumption contributes significantly to the ecological requirements, namely to save energy and increase overall system utilization. The focus here is to integrate the functional, operational and energy awareness into the systems providing autonomous functioning with reduced energy consumption. The rationale, expressing power and practical value of the approach are illustrated on e-mobility and cloud computing application domains. The two complex domains appear to be fairly different. However, taking a

closer look at the requirements of the two scenarios it becomes noticeable that the problem domains share numerous generic system properties, especially seen from the optimized control perspective.

The paper presents work in progress focusing on energy optimization in complex distributed control systems. It further elaborates methods and techniques to model and construct complex distributed systems with service components and ensembles. The rationale of the approach is presented through close requirements analysis, system modeling and development. The deployment is illustrated by the science cloud application scenario. Finally, the approach is summarized giving further directions for the work to come.

## II. REQUIREMENTS ANALYSIS

To explore the system requirements, two complex application domains are closely examined: e-mobility control and cloud computing.

E-mobility is a vision of future transportation by means of electric vehicles network allowing people to fulfill their individual mobility needs in an environmental friendly manner (decreasing pollution, saving energy, sharing vehicles, etc).

Cloud computing is an approach that delivers computing resources to users in a service-based manner, over the internet, thus re-enforcing sharing and reducing energy consumption).

At a first glance electric vehicular transportation and distributed computing on demand have nothing really in common!

### A. Common Characteristics

In a closer examination the two systems, though very different, have a number of common characteristics.

#### 1) Massive Distribution and Individual Interest

E-mobility deals with managing a huge number of e-vehicles that transport people from one place to another taking into account numerous restrictions that the electrical transportation means imposes.

Each cloud computing user has also his/her individual application demands and interest to efficiently execute it on the cloud. The goal of cloud computing is to satisfy all these competing demands.

Both applications are characterized with huge number of single entities with individual goals.

2) *Sharing and Collectiveness*

In order to cover longer distances, an e-vehicle driver must interrupt the journey to either exchange or re-charge the battery. Energy consumption has been the major obstacle in a wider use of electric vehicles. Alternative strategy is to share e-vehicles in a way that optimizes the overall mobility of people and the spending of energy. In other words: when my battery is empty – you will take me further if we go in the same direction and vice versa [8].

The processing statistics show that most of the time computers are idle – waiting for input to do some calculations. Computers belong amongst the fastest yet most wasteful devices man has ever made. And they dissipate energy too. Cloud computing overcomes that problem by sharing computer resources making them better utilized. In another words, if my computer is free – it can process your data and vice versa; or even better, let us have light devices and leave a heavy work for the cloud [9].

At a closer look “sharing and collectiveness” are common characteristics of both application domains!

3) *Awareness and Knowledge*

E-mobility can support coordination only if e-vehicles know their own restrictions (battery state), destinations of users, re-charging possibilities, parking availabilities, the state of other e-vehicles nearby. With such knowledge collective behavior may take place, respecting individual goals, energy consumption and environmental requirements.

Cloud computing deals with dynamic (re-)scheduling of available (not fully used) computing resources. Maximal utilization can only be achieved if the cloud is “aware” of the users’ processing needs and the states of the deployed cloud resources. Only with such knowledge a cloud can make a good utilization of computers while serving individual users’ needs.

At a closer look “awareness” of own potentials, restrictions and goals as well as those of the others is a common characteristic. Both domains require self-aware, self-expressive and self-adaptive behavior based on a knowledge about those “self\*” properties.

4) *Dynamic and Distributed Energy Optimization*

E-mobility is a distributed network that manages numerous independent and separate entities such as e-vehicles, parking slots, re-charge stations, drivers. Through collective and awareness-rich control strategy the system may dynamically re-organize and optimize the use of energy while satisfying users’ transportation needs.

Cloud computing actually behaves as a classical distributed operating system with a goal to maximize operation and throughput and minimize energy consumption, performing tasks of multiple users.

At a closer look “dynamic and distributed optimization” is inherent characteristic of the control environment for both application domains.

TABLE I. COMMON CHARACTERISTICS

<i>Common feature</i>	<i>Cloud computing</i>	<i>E-Mobility</i>
Single entity	Computing resource	Vehicle, driver, park place, charging station
Individual goal	Efficient execution	Individual route plan
Ensemble	application , cpu pool,	Free vehicles, free park places, etc
Global goal	Resource availability, optimal throughput	Travel, journey, low energy
Self-awareness	avail-able resources; computational requirements, etc	Awareness of own state and restrictions
Autonomous and collective behavior	Decentralized decision making, global optimization	Reaching all destinations in time, minimizing costs
Optimization	Availability, computational task execution	Destination achievement in time, vehicle/infrastructure usage
Adaptation	According to available resources	According to traffic, individual goals, infrastructure, resource availability
Robustness	Failing resources	Range limitation, charging battery infrastructure resources

B. *Common Approach*

This set of common features serve as a basis for modeling of such systems leading to a generic framework for developing and deploying complex autonomic systems. The table 1 summarized the common requirements that lead to four major behavioral principles: adaptation, self-awareness, knowledge and emergence.

III. MODELING

Control systems for the two application domains have many common characteristics: they are highly collective, constructed of numerous independent entities that share common goals. Their elements are both autonomous and cooperative featuring a high level of self-awareness and self-expressiveness. A complex control system built out of such entities must be robust and adaptive offering maximal utilization with minimal energy and resource use.

Formal specification, programming and controlling of a complex massively parallel distributed system that features awareness, autonomous and collective behavior, adaptive optimization and robust functioning are grand challenges of computer science. These challenges, present in most of complex control systems, have served as motivation and inspiration for this approach [7].



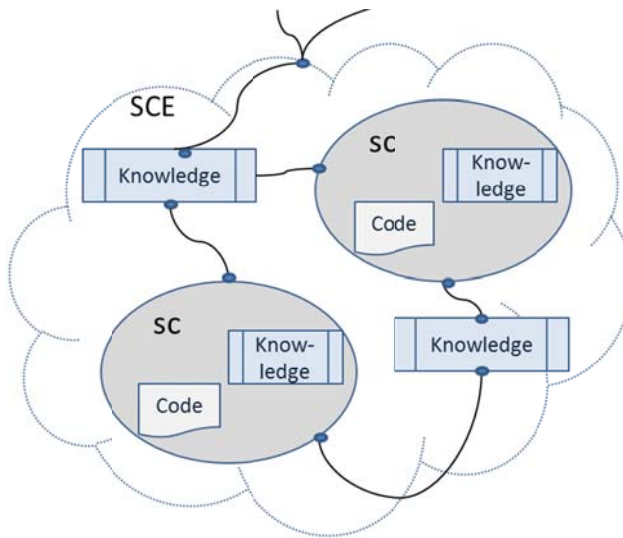


Figure 1. Service components and their ensembles

A complex system is decomposed in service components - major individual entities, and service component ensembles - composition structures that reflect communication and joint needs of service components:

- SC – service component are single system entities that have their requirements and functionality, usually representing their individual goals,
- SCE – service component ensembles are collections of service components usually representing collective system goals (as means to dynamically structure independent and distributed system entities).

The system structuring is depicted on Fig. 1.

Both components and ensembles have knowledge elements used to express their state and requirements. Based on this declarative knowledge, awareness, emergence and adaptive behavior can be achieved [7].

Fig. 2 illustrates an abstract view of modeling massively distributed systems with service components and ensembles. At the first level the real system entities are presented with different symbols representing different types of components. At the upper levels, different groupings are illustrated where components can be linked in ensembles, according to their requirements. There may be different ways of grouping, represented by different ensemble levels. One component can be a member of different ensembles at the same time. Ensembles are not fixed, during the system life time and according to the on-going states, re-grouping happens as a system response to dynamic changes.

*A. Modeling e-Mobility with Ensembles*

Applying the general modeling strategy as depicted on Fig. 2 to e-mobility scenario, the different symbols at the first level could be interpreted as (1) users, (2) e-vehicles, (3) charging stations and (4) park places service components, where each component has knowledge on its

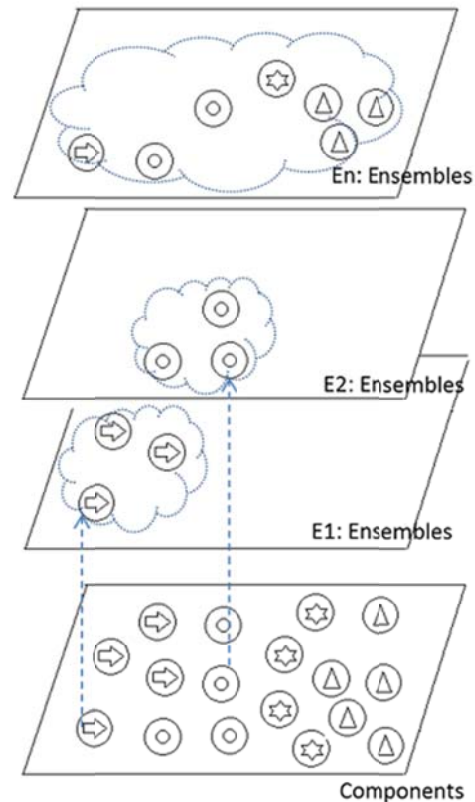


Figure 2. Modeling with components and ensembles

own state and needs. A user component knows the route plan having a goal to reach different places in a given time. A vehicle component has knowledge about its occupancy and battery state. Park places and charging stations maintain their availability/reservation plan. These major service components of the e-mobility scenario build the individual types with a huge number of instances. The E1 and E2 ensemble levels show grouping according to the service component types, allowing users with nearby destinations to form an ensemble (with a common goal to reach the same destination and a possibility to share the vehicle) or vehicles with fully charged batteries at the same location to form an ensemble of available vehicles. The “En” ensemble level shows the e-mobility application with one user planning to use two vehicles, one parking place and a number of possible charging stations.

*B. Modeling Cloud Computing with Ensembles*





In a similar manner, the same model shown on the Fig. 2 may represent an abstract cloud computing scenario. The major system elements represented by different symbols at the first level (E1) could be interpreted as (1) user applications, (2) remote computer CPUs, (3) local memory and (4) local application service components. Thereby, each component has knowledge about its own state and requirements. A user application component knows the requests for execution (in terms of CPU, minimal space,

etc.). A remote computer component has knowledge about its processing capabilities and a current utilization. Disk components have knowledge of their capacity. Appi components have descriptions of the available apps at the local computer.

The E1 and E2 ensemble levels show grouping according to the service component types, allowing e.g. grouping of appis of the same type with similar requests to form an ensemble or different CPUs to form an ensemble of available CPUs. The “En” ensemble level shows the cloud application with one user appi running at one remote CPU with a possibility to migrate to another CPU (with similar configuration), using one memory resource with a possibility to access a number of local applications.

Table 2 summarizes major service components within both application scenario mapping.

TABLE II. MAJOR SERVICE COMPONENTS

Symbols	E-Mobility	Cloud computing
	Users	User applications
	Electric vehicles	Remote computer CPUs
	Charging stations	Local memory
	Park places	Local application services

C. SCEL Language Programming Abstractions

The challenge for developers of complex distributed systems is to find proper linguistic abstractions to cope with individual vs. collective requirements of system elements and their need to respond to dynamic changes in an autonomous manner. A set of semantic constructs has been proposed [10,11] that represent behaviors, knowledge and composition supporting programming of awareness-rich system.

The basic ingredient of SCEL - Software Component Ensemble Language is the notion of autonomic component  $I[K; \Pi; P]$  that consists of:

- An interface  $I$  in a form of attributes – visible to other components.
- Knowledge repository  $K$  managing information about component interface, requirements, major state attributes etc. Managing such knowledge allows for self-aware behavior and dynamic interlinking with other system components.
- A set of policies  $\Pi$  that manage the internal and external interaction.
- A set of process  $P$  defines component functionality specific to both the application and internal management of knowledge, polices and communication.

The structure and organization of the SCEL notation is illustrated in Fig. 3,

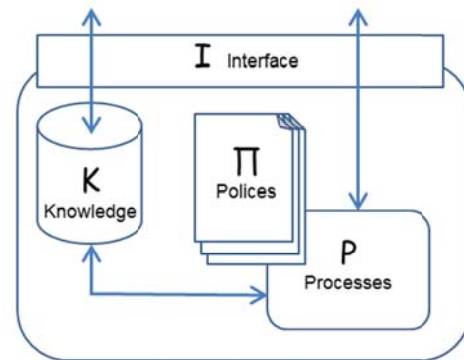


Figure 3. SCEL elements

Systems:  $S ::= C \mid S_1 \parallel S_2 \mid (\nu n)S$   
 Components:  $C ::= I[K, \Pi, P]$   
 Processes:  $P ::= \text{nil} \mid a.P \mid P_1 + P_2 \mid P_1[ P_2 ] \mid X \mid A(p)$   
 Actions:  $a ::= \text{get}(T)@c \mid \text{qry}(T)@c \mid \text{put}(t)@c \mid \text{new}(I, K, \Pi, P)$   
 Targets:  $c ::= n \mid x \mid \text{self} \mid P \mid I.p$

The code above shows a fraction of SCEL syntax (with notation for  $S$  - systems,  $C$  - components,  $P$  - processes,  $a$  - actions and  $c$  - targets); a fully detailed presentation of SCEL syntax and semantics can be found in [10, 11].

The SCEL aggregates both semantics and syntax power to express autonomic behavior. At one side, being abstract and rigorous SCEL allows for formal reasoning about system behavior, at another, it needs further programming tools to support system development and deployment. Formal reasoning, modeling and validation are covered in referenced articles about SCEL. Here, the focus is more on pragmatic orientation on a given application scenario.

IV. DEVELOPING AND DEPLOYING AUTONOMOUS SYSTEMS

A way from high level modeling to development and deployment of software intensive systems is a complex endeavor. Reasoning and validation often require high-level abstractions, while implementation calls for detailed programming and low-level deployments. To bridge this gap a number of intermediate tools are being developed that assist in the engineering process [7,12].

A. Java Framework for SCEL Programming and Model Checking

To execute SCEL programs, the jRESP framework has been developed. This is a Java runtime environment providing means to develop autonomic and adaptive systems programmed in SCEL [13]. By relying on the jRESP API, a programmer can embed the SCEL paradigm in Java applications.

A prototype statistical model-checking running on top of jRESP simulation environment has been implemented. Following this approach, a randomized algorithm is used to verify whether the implementation of a system satisfies a specific property with a certain degree of confidence. The statistical model-checker is parameterized with respect to a given tolerance  $t$  and error probability  $p$ . The used algorithm guarantees that the difference between the computed values and the exact ones is greater than  $t$  with a probability lower than  $p$ .

The model-checker included in jRESP can be used to verify reachability properties. These properties allow one to evaluate the probability to reach, within a given deadline, a configuration where a given predicate on collected data is satisfied [13].

**B. Developing Science Cloud**

Cloud computing is a modern paradigm for programming and utilizing distributed infrastructure resources in a dynamic way. Cloud-based systems are safety- and security-critical systems; they need to satisfy time-critical performance-based quality of service properties and to dynamically adapt to changes in the potentially hostile and uncertain environment they operate in. These aspects make distributed cloud-based systems complex and hard to design, build, test, and verify. The cloud scenario taken here is the cloud as a platform with voluntary peer-to-peer configuration, meant to execute scientific applications [9]. It closely followed the modeling approach described in previous sections.

**1) Service Components**

Each instance of the Science Cloud Platform (SCP), running on a physical or virtual machine is considered to be a service component in the previous described sense. Fig. 4 shows the functionality required by a Science Cloud Platform instance. Two major characteristics of SCPs are further explored: knowledge and connectivity.

**2) Knowledge**

Each SCPi has knowledge consisting of (1) its own properties (set by developers), (2) its infrastructure (CPU load, available memory), and (3) other SCPis (acquired through the network). Since there is no global coordinator, each SCPi must build its own view and act upon the available knowledge. The SCPi may acquire knowledge about its infrastructure using an infrastructure sensing plugin which provides information about static values, such as processor speed, available memory, available disk space, number of cores etc. and dynamic values, such as currently used memory, disk space, or CPU load.

SCPi properties are important when specifying conditions (Service Level Agreements, SLAs) for the applications. For example, when looking for a new SCPi to execute an application, low latency between the SCPs might be interesting. Other requirements may be harder: For example, an application may simply not fit on an SCPi

because of the lack of space whereas another may require a certain amount of memory.

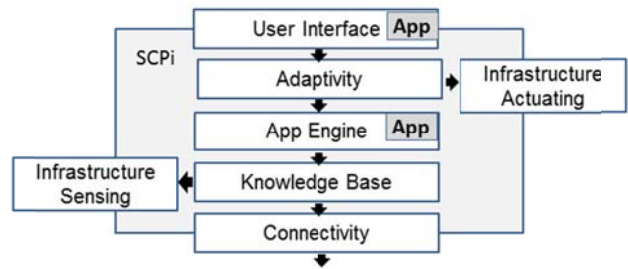


Figure 4. Science Cloud Framework

**3) Connectivity**

Each SCPi has a connectivity component which enables it to talk to other SCPs over the network. The protocol followed by these communications must enable SCPs to find one another and establish links, for example by manually entering a network address or by a discovery mechanism. Furthermore, SCPs must be able to query others for knowledge and at the same time distribute their own knowledge. Finally, the protocol must support exchange of data and applications.

Fig. 4 illustrates an instance of a science cloud platform as a part of a rich virtual framework for executing scientific applications. Through awareness of its own properties and those of others it offers maximal utilization of available computing resources within the cloud.

As already indicated an SCPi is adaptive and can react to conditions such as overload, shutdown of other SCPs, etc. Furthermore, it must watch over the apps executed and guarantee their SLAs (Service Level Agreement) [7,9]. This functionality is performed in an adaptivity logic component. The adaptivity logic is exchangeable, application-independent, and has a direct relation to the SLAs of applications. The adaptivity logic itself can be written in a standard programming language or custom domain-specific languages or rules.

Finally, each SCPi provides the application execution service to upper levels. The applications run on the platform must implement some API for the platform to be able to work with them (i.e. starting, stopping, working with data, etc.). One example of an app is the data storage service which allows users to store data in the cloud.

**4) Ensembles**

A Science Cloud Platform Ensemble (SCPe) consists of individual SCPs based on a set of properties of the SCPs and/or the SLAs of applications. In another words, an ensemble consists of SCPs which work together to run one application in a fail-safe manner and under consideration of the SLA of that application, which may require a certain number of SCPs, certain latency between the parts, or have restrictions on processing power or on memory.

At runtime, an ensemble may gain new SCPs or lose them depending on the behavior of the SCPis themself and

also on the load generated by the application itself or other applications running on the SCPIs.

### C. Application Deployment

Currently, a prototype of a science cloud platform is being developed and tested in a physical network connecting two universities [7]. The experimental platform does feature ad hoc and voluntary behavior supporting dynamic re-configuration of physical layers and application migration on an upper level. High-level SCEL modeling and model checking provide formal means for properties proofs while a prototype implementation offers pragmatic means to test deployment and effectiveness of autonomous and self-aware behavior.

## V. CONCLUSION

The paper presents a unified approach to model, validate and deploy complex distributed systems with massive number of nodes that respect both individual and global goals. Non-centralized character of the approach allows for autonomic and self-aware behavior, which is achieved by introduction of knowledge elements and enrichment of compositional and communication primitives with awareness of both system requirements and individual state of the computing entities.

The essence of the approach is to de-compose a complex system into a number of generic components and to further compose the system into ensembles of service components.

The inherent complexity of ensembles is a huge challenge for developers. Thus, the whole system is decomposed into well-understood building blocks, reducing the innumerable interactions between low-level components to a manageable number of interactions between these building blocks. The result is a so-called hierarchical ensemble, built from service components, simpler ensembles and knowledge units connected via a highly dynamic infrastructure. Ensembles exhibit four main characteristics: adaptation, self-awareness, knowledge and emergence, yielding a sound technology for engineering autonomous systems [5,7]. A number of linguistic constructs and validation and programming tools are under development and are being tested in different application scenarios.

This paper presents an integrated view (from high level modeling to application deployment) of a complex approach which has been described by a number of referenced papers, each focusing on different aspects of the work: SCEL modeling [10,11] and system validation [13], adaptation aspects[8], knowledge management and deployments [8,9] and engineering aspects [5,7]. Further contribution of this paper is in optimized control based on awareness and autonomous behavior.

Optimized distributed control with improved throughput and utilization of the cloud and e-mobility frameworks

contribute significantly to the overall strategy to reduce energy consumption. Sharing principle instead of exclusive use of the computing and transportation means represent a significant challenge (requiring significant changes in our perception of vehicles and computers) in the application domains under consideration. This principle will undoubtedly play an important role in extending the application domains.

### ACKNOWLEDGMENT

Most of the work presented here has been done under the ASCENS project (project number FP7- 257414) [7], funded by the European Commission within the 7th Framework Programme, pervasive adaptation initiative. Special thanks go to the developers of SCEL language (Rocco De Nicola from IMT Lucca and his group) and the developers of cloud computing application (Philip Mayer and the whole ASCENS team from LMU Munich).

### REFERENCES

- [1] Project InterLink: <http://interlink.ics.forth.gr> [retrieved: Feb.2013].
- [2] I. Sommerville et al., "Large-scale complex it systems". Commun. ACM, vol.55, no.7, 2012, pp.71-77.
- [3] M. Hoelzl, A. Rauschmayer, and M. Wirsing, "Engineering of software-intensive systems", in Wirsing, M., Banatre, J.P., Hoelzl, M., Rauschmayer, A., eds.: Software-Intensive Systems and New Computing Paradigms. Vol.5380 of LNCS, 2008, pp.1-44.
- [4] L. Xu, G. Tan, X. Zhang, and J. Zhou, "Energy aware cloud application management in private cloud data center", 2011 International Conference on Cloud and Service Computing, 2011, pp.274-279.
- [5] C. Seo, "Energy-Awareness in Distributed Java-Based Software Systems", 21st IEEE International Conference on Automated Software Engineering (ASE'06), 2006, pp.343-348.
- [6] M. Hoelzl and M. Wirsing, "Towards a system model for ensembles", in G. Agha, O. Danvy, and J. Meseguer (eds.), Formal Modeling: Actors, Open Systems, Biological Systems, Lecture Notes in Computer Science Vol.7000, 2011, pp. 241-261.
- [7] Project ASCENS (Autonomic Service-Component Ensembles), <http://www.ascens-ist.eu> ASCENS, 2012, [retrieved: Feb.2013].
- [8] D. Abeywickrama, F. Zambonelli, and N. Hoch, "Towards Simulating Architectural Patterns for Self-Aware and Self-Adaptive Systems", In 2nd SASO Workshop on Awareness in Autonomic Systems, Lyon (F), 2012, pp.87-94.
- [9] P. Zormeier, A. Klarl, C. Kroiss, and P. Mayer, "Science Cloud: Modelling and Implementing the Peer-to-Peer DHT protocol 'Chord'", Technical Report, Ludwig-Maximilians-Universitt Mnchen, Germany, 2012.
- [10] R. De Nicola, G-L. Ferrari, M. Loreti, and R. Pugliese, "A Language-based Approach to Autonomic Computing", In Formal Methods for Components and Objects, vol.7542 Lecture Notes in Computer Science, 2012, pp.26-48.
- [11] R. De Nicola, M. Loreti, R.Pugliese, and F. Tiezzi, "SCEL: a language for autonomic computing", Technical Report, [retrieved: Feb.2013].
- [12] M.P. Ashley-Rollman, S.C. Goldstein, P. Lee, T.C. Mowry, and P. Pillai, "Meld: A declarative approach to programming ensembles", In: IROS, IEEE, 2007, pp.2794-2800.
- [13] M. Loreti. jRESP: a run-time environment for scel programs, Technical Report, <http://rap.dsi.unifi.it/scel/>, [retrieved: Feb.2013].

# Comparison Between Self-Stabilizing Clustering Algorithms in Message-Passing Model.

Mandicou BA, Olivier FLAUZAC, Rafik MAKHLOUFI, Florent NOLOT  
 Université de Reims Champagne-Ardenne  
 CReSTIC - SysCom (EA 3804)  
 Reims, France  
 {mandicou.ba, olivier.flauzac, rafik.makhloufi, florent.nolot}@univ-reims.fr

Ibrahima NIANG  
 Université Cheikh Anta Diop  
 Laboratoire d'Informatique de Dakar (LID)  
 Dakar, Sénégal  
 iniang@ucad.sn

**Abstract**—Most Ad Hoc networks use diffusion to communicate. This approach requires many messages and may cause network saturation. To optimize these communications, one solution consists in structuring networks into clusters. In this paper, we present a new self-stabilizing asynchronous distributed algorithm based on message-passing model. We compare the proposed algorithm with one of the best existing solutions based on message-passing model. Our approach does not require any initialization and builds non-overlapping k-hops clusters. It is based only on information from neighboring nodes with periodic messages exchange. Starting from an arbitrary configuration, the network converges to a stable state after a finite number of steps. A legal configuration is reached after at most  $n + 2$  transitions and uses at most  $n * \log(2n + k + 3)$  memory space, where  $n$  is the number of network nodes. Using the *OMNeT++* simulator, we performed an evaluation of the proposed algorithm to notably show that we use fewer messages and stabilizing time is better.

**Keywords**—ad hoc networks; clustering; distributed algorithms; self-stabilizing; *OMNeT++* simulator

## I. INTRODUCTION

In Ad Hoc networks, the most frequently used communication solution is diffusion. This is a simple technique that requires few calculations. But this method is expensive and may cause network saturation. In order to optimize this communication, which is an important source of resource consumption, one solution is to structure the network in *trees* [1] or *clusters* [2].

Clustering consists in organizing the network into groups of nodes called clusters, thus giving a hierarchical structure [3]. Each cluster is managed by a particular node called clusterhead. A node is elected clusterhead using a metric such as the mobility degree, node's identity, node's density, etc. or a combination of these parameters. Several solutions of clustering have been proposed. They are classified into 1-hop and k-hops algorithms. In 1-hop solutions [4], [5], [6], [7] nodes are at a distance of 1 from the clusterhead and the maximum diameter of clusters is 2. However, in k-hops solutions [8], [9] nodes can be located at a distance of  $k$  from the clusterhead and the maximum diameter of clusters is  $2k$ . However, these approaches, generate a lot of traffic and require considerable resources.

In this paper, we propose a self-stabilizing asynchronous distributed algorithm that builds k-hops clusters. Dijkstra de-

vised a distributed system to be self-stabilizing if, regardless of the initial state, the system is guaranteed to reach a legitimate (correct) state in a finite time [10]. Our approach builds non-overlapping k-hops clusters and does not require initialization. It is based on the criterion of maximum identity attached to the nodes for clusterhead selection and relies only on the periodic exchange of messages with the 1-hop neighborhood. The choice of the identity metric provides more stability against dynamic criteria such as mobility degree and weight of nodes.

The remainder of the paper is organized as follows. In Section II, we describe some related works of self-stabilizing clustering solutions. Section III presents our contribution. In Section IV, we describe the computational model used in the paper and give some additional concepts. In Section V, we first present a broad and intuitive explanation of the algorithm before defining it more formally. In Section VI, we present performance evaluation conducted with the *OMNeT++* simulator. Finally, we conclude and present some future work in Section VII.

## II. RELATED WORK

Several clustering solutions have been done in the literature [4], [5], [6], [7], [8], [11], [9]. Approaches [6], [7], [11], [9] are based on state model at opposed of message-passing model algorithms [4], [5], [8].

Self-stabilizing algorithms presented in [4], [5], [6], [7] are 1-hop clustering solutions.

A metric called *density* is used by Mitton et al. in [4], in order to minimize the reconstruction of structures for low topology change. Each node calculates its density and broadcasts it to its neighbors located at 1-hop. For the maintenance of clusters, each node calculates periodically its mobility and density.

Flauzac et al. [5], have proposed a self-stabilizing clustering algorithm, which is based on the identity of its neighborhood to build clusters. This construction is done using the identities of each node that are assumed unique. The advantage of this algorithm is to combine in the same phase the neighbors discovering and the clusters establishing. Moreover, this deterministic algorithm constructs disjoint clusters, i.e., a node is always in only one cluster.

In [6], Johnen et al. have proposed a self-stabilizing protocol designed for the state model to build 1-hop clusters whose size is bounded. This algorithm guarantees that the network nodes are partitioned into clusters where each one has at most *SizeBound* nodes. The clusterheads are chosen according to their *weight* value. In this case, the node with the higher weight becomes clusterhead. In [7], Johnen et al. have extended the proposal from [6]. They have proposed a robust self-stabilizing weight-based clustering algorithm. The robustness property guarantees that, starting from an arbitrary configuration, after one asynchronous round, the network is partitioned into clusters. After that, the network stays partitioned during the convergence phase toward a legitimate configuration where clusters verify the ad hoc clustering properties.

Self-stabilizing algorithms proposed in [8], [11], [9] are k-hops clusters solutions.

In [11], using criterion of minimal identity, Datta et al. have proposed a self-stabilizing distributed algorithm designed for the state model that computes a subset  $D$  is a minimal  $k$ -dominating set of graph  $G$ . Using  $D$  as the set of *clusterheads*, a partition of  $G$  into clusters, each of radius  $k$ , follows. This algorithm converges in  $O(n)$  rounds and  $O(n^2)$  steps and requires  $\log(n)$  memory space per process, where  $n$  is the size of the network.

Datta et al. [9], using an arbitrary metric, have proposed a self-stabilizing  $k$ -clustering algorithm base on a state model. Note that  $k$ -clustering of a graph is a partition of nodes into disjoint clusters in which every node is at a distance of at most  $k$  from the *clusterhead*. This algorithm executes in  $O(nk)$  rounds and requires  $O(\log(n) + \log(k))$  memory space per process, where  $n$  is the network size.

In [8], Miton et al. applied self-stabilization principles over a clusterization protocol proposed in [4] and presents properties of robustness. Each node calculates its density and broadcasts it to its neighbors located at k-hops. This robustness is an issue related to the dynamicity of ad hoc networks, to reduce the time stabilization and to improve network stability.

### III. CONTRIBUTION

We propose a self-stabilizing asynchronous distributed algorithm that builds k-hops clusters. Our approach is based on a message-passing model as opposed to solutions proposed in [6], [7], [11], [9]. We use the criterion of maximum identity that brings more stability compared to metric variables used in [8], [4], [6], [7]. Our algorithm structures the network into non-overlapping clusters with a diameter at most equal to  $2k$ . This structuring does not require any initialization. It is based only on information from neighboring nodes. Contrary to other clustering algorithms, we use a unique message to discover the neighborhood of a node at distance 1 and to structure the network into  $k$ -hops clusters. Starting from an arbitrary configuration, the network converges to a legal configuration after a finite number of steps. A legal configuration is reached after at most  $n + 2$  transitions and requires at most  $n * \log(2n + k + 3)$  memory space, where  $n$  is

the number of network nodes. Using the *OMNeT++* simulator, we performed an evaluation of the proposed algorithm and a comparison with one of the best existing solution based on message-passing model [8]. We show that we use less messages than and stabilizing time is better.

### IV. MODEL

We consider our network as a distributed system that can be modeled by an undirected graph  $G = (V, E)$ .  $V = n$  is the set of network nodes and  $E$  represents all existing connections between nodes. An edge  $(u, v)$  exists if and only if  $u$  can communicate with  $v$  and vice-versa. This means that all links are bidirectional. In this case, the nodes  $u$  and  $v$  are neighbors. The set of neighbors  $v \in V$  of node  $u$  is marked  $N_u$ . Each node  $u$  of the network has a unique identifier  $id_u$  and can communicate with  $N_u$ . We define the distance  $d_{(u,v)}$  between nodes  $u$  and  $v$  in the graph  $G$  as the minimum number of edges along the path between  $u$  and  $v$ .

Our algorithm is designed for the asynchronous message-passing model following standard models for distributed systems given in [12], [13]. For this purpose, each pair of nodes is connected by a bi-directional link. Links are asynchronous and messages transit time is finite but not bounded. Moreover links are reliable. They do not create, corrupt or lose messages. Furthermore, each node  $u$  periodically sends to its neighbors a message that is received correctly within some finite but unpredictable time by all its 1-hop neighbors. Each node  $u$  maintains a table containing the current state of its neighbors at distance 1. Upon receiving a message, a node  $u$  executes our clustering algorithm.

### V. SELF-STABILIZING K-HOPS CLUSTERING ALGORITHM

#### A. Preliminaries

We give some definitions used in the following.

**Definition 5.1:** (Cluster) We define a  $k$ -hops cluster as a connected graph in the network, with a diameter less than or equal to  $2k$ . The set of all the nodes of a cluster  $i$  is denoted  $V_i$ .

**Definition 5.2:** (Cluster identifier) Each cluster has a unique identifier corresponding to the greatest node identity in its cluster. The identity of a cluster that owns a node  $u$  is denoted  $cl_u$ .

In our clusters, each node  $u$  has a status noted  $status_u$ . Thus, a node can be clusterhead (*CH*), a *Simple Node* (*SN*) or a *Gateway Node* (*GN*). Moreover, each node selects a neighbor  $v \in N_u$ , noted  $gn_u$ , through which it passes to reach its *CH*.

**Definition 5.3:** (Nodes status)

- **Clusterhead (CH):** a node  $u$  has *CH* status if it has the highest ID among all nodes of its cluster.
  - $status_u = CH \iff \forall v \in V_{cl_u}, (id_u > id_v) \wedge (dist_{(u,v)} \leq k)$ .
- **Simple Node (SN):** a node  $u$  has *SN* status if  $u$  and all its neighbors are in the same cluster.
  - $status_u = SN \iff (\forall v \in N_u, cl_v = cl_u) \wedge (\exists w \in V_u / (status_w = CH) \wedge (dist_{(u,w)} \leq k))$ .

- **Gateway Node (GN):** a node  $u$  has  $GN$  status if exists a node  $v$  in neighborhood in a different cluster.
  - $status_u = GN \iff \exists v \in N_u, (cl_u \neq cl_v)$ .

**Definition 5.4:** (Node Coherence)

A node  $u$  is *coherent node* if and only if it is in one of the following states:

- if  $status_u = CH$  then  $(cl_u = id_u) \wedge (dist_{(u, CH_u)} = 0) \wedge (gn_u = id_u)$ ,
- if  $status_u \in \{SN, GN\}$  then  $(cl_u \neq id_u) \wedge (dist_{(u, CH_u)} \neq 0) \wedge (gn_u \neq id_u)$ .

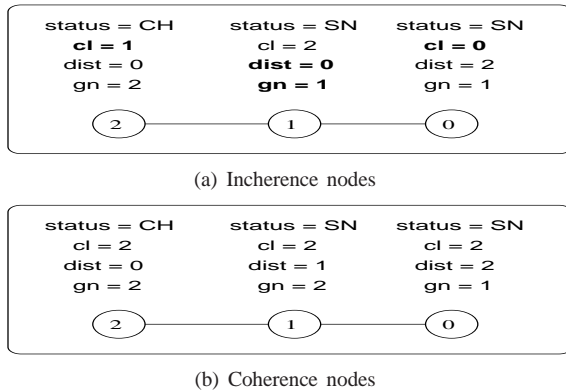


Fig. 1. Coherent and incoherent nodes

**Definition 5.5:** (Node stability)

A node  $u$  is *stable node* if and only if its variables no longer change, it is coherent and satisfies the following states:

- if  $status_u = CH$  then  $\forall v \in N_u, (status_v \neq CH) \wedge \{((cl_v = cl_u) \wedge (id_v < id_u)) \vee ((cl_v \neq cl_u) \wedge (dist_{(v, CH_v)} = k))\}$ . (Example of node 9 in cluster  $V_9$ )
- if  $status_u = SN$  then  $\forall v \in N_u, (cl_v = cl_u) \wedge (dist_{(u, CH_u)} \leq k) \wedge (dist_{(v, CH_v)} \leq k)$ . (Example of node 0 in cluster  $V_{10}$  or node 7 in  $V_9$ ).
- if  $status_u = GN$  then  $\exists v \in N_u, (cl_v \neq cl_u) \wedge \{((dist_{(u, CH_u)} = k) \wedge (dist_{(v, CH_v)} \leq k)) \vee ((dist_{(v, CH_v)} = k) \wedge (dist_{(u, CH_u)} \leq k))\}$ . (Example of node 2 in cluster  $V_9$  or node 8 in  $V_{10}$ ).

**Definition 5.6:** (Network stability)

The network is *stable* if and only if all nodes are stable nodes. (see figure 2)

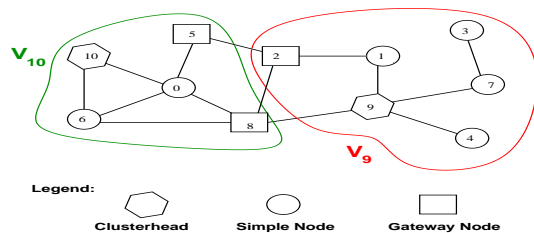


Fig. 2. Stable Nodes - Stable network

### B. Basic Idea of Our Solution

Our algorithm is self-stabilizing and does not require any initialization.

Starting from any arbitrary configuration, with only one type of message exchanged, the nodes are structured in non-overlapping clusters in a finite number of steps. This message is called *hello message* and it is periodically exchanged between each neighbor nodes. It contains the following four items information: node identity ( $id_u$ ), cluster identity ( $cl_u$ ), node status ( $status_u$ ) and the distance to clusterhead  $dist_{(u, CH_u)}$ . Note that cluster identity is also the identity of the clusterhead. Thus, the hello message structure is  $hello(id_u, cl_u, status_u, dist_{(u, CH_u)})$ . Furthermore, each node maintains a neighbor table  $StateNeigh_u$  that contains the set of its neighboring nodes states. Whence,  $StateNeigh_u[v]$  contains the states of nodes  $v$  neighbor of  $u$ .

The solution that we propose proceeds as follows:

As soon as a node  $u$  receives a hello message, it executes Algorithm 1. During this algorithm,  $u$  executes these three steps consecutively. The first step is to update neighborhood, the next step is to manage the coherence and the last step is to build the clusters. After this three steps,  $u$  sends a hello message to its neighbors.

After updating the neighborhood, nodes check their coherence. For example, as a clusterhead has the highest identity, if a node  $u$  has  $CH$  status, its cluster identity must be equal to its identity. In Figure 1(a), node 2 is clusterhead. Its identity is 2 and its cluster identity is 1, so node 2 is not a coherent node. Similarly for nodes 1 and 0, because they do not satisfy definition 5.4. Each node detects its incoherence and corrects its during the coherence management step. Figure 1(b) shows nodes that are coherent.

During the clustering step, each node compares its identity with those of its neighbors at distance 1. A node  $u$  elects itself as a clusterhead if it has the highest identity among all nodes of its cluster. If a node  $u$  discovers a neighbor  $v$  with a highest identity then it becomes a node of the same cluster as  $v$  with  $SN$  status. If  $u$  receives again a hello message from another neighbor that is in another cluster than  $v$ , the node  $u$  becomes gateway node with  $GN$  status. As the hello message contains the distance between each node  $u$  and its clusterhead,  $u$  knows if the diameter of cluster is reached. So it can choose another cluster.

### C. k-hops self-stabilizing algorithm

Each node  $u$  of the network knows the  $k$  parameter value and executes the Algorithm 1.

## VI. PERFORMANCE ANALYSIS

In [14], we have proved that our network is stable after at most  $n + 2$  transitions and requires at most  $n * \log(2n + k + 3)$  memory space. This reflects the worst case scenario of a topology where the nodes form an ordered chain. Ad Hoc networks are often characterized by random topologies.

In order to evaluate the average performance of our solution in a random topology, we have implemented our algorithm using the *OMNeT++* environment simulation [15]. For generating random graphs, we have used *SNAP* library [16]. All simulations were carried out using *Grid'5000* platform [17].

**Algorithm 1: k-hops clustering algorithm**

```

/* Upon receiving message from a neighbor */
Predicates
 $P_1(u) \equiv (status_u = CH)$ 
 $P_2(u) \equiv (status_u = SN)$ 
 $P_3(u) \equiv (status_u = GN)$ 
 $P_{10}(u) \equiv (cl_u \neq id_u) \vee (dist_{(u,CH_u)} \neq 0) \vee (gn_u \neq id_u)$ 
 $P_{20}(u) \equiv (cl_u = id_u) \vee (dist_{(u,CH_u)} = 0) \vee (gn_u = id_u)$ 
 $P_{40}(u) \equiv$ 
 $\forall v \in N_u, (id_u > id_v) \wedge (id_u \geq cl_v) \wedge (dist_{(u,v)} \leq k)$ 
 $P_{41}(u) \equiv \exists v \in N_u, (status_v = CH) \wedge (cl_v > cl_u)$ 
 $P_{42}(u) \equiv \exists v \in N_u, (cl_v > cl_u) \wedge (dist_{(v,CH_v)} < k)$ 
 $P_{43}(u) \equiv \forall v \in N_u / (cl_v > cl_u), (dist_{(v,CH_v)} = k)$ 
 $P_{44}(u) \equiv \exists v \in N_u, (cl_v \neq cl_u) \wedge \{(dist_{(u,CH_u)} = k) \vee (dist_{(v,CH_v)} = k)\}$ 

Macros
 $NeighCH_u = \{id_v / v \in N_u \wedge status_v = CH \wedge cl_u = cl_v\}$ 
 $NeighMax_u =$ 
 $(Max\{id_v / v \in N_u \wedge status_v \neq CH \wedge cl_u = cl_v\} \wedge$ 
 $(dist_{(v,CH_u)} = Min\{dist_{(x,CH_u)}, x \in N_u \wedge cl_x = cl_v\}))$ 

Rules
/* Update neighborhood */
 $StateNeigh_u[v] := (id_v, cl_v, status_v, dist_{(v,CH_v)});$ 

/* Cluster-1: Coherent management */
 $R_{10}(u) :: P_1(u) \wedge P_{10}(u)$ 
 $\rightarrow cl_u := id_u; gn_u := id_u; dist_{(u,CH_u)} = 0;$ 
 $R_{20}(u) :: \{P_2(u) \vee P_3(u)\} \wedge P_{20}(u) \rightarrow$ 
 $status_u := CH; cl_u := id_u; gn_u := id_u; dist_{(u,CH_u)} = 0;$ 

/* Cluster-2: Clustering */
 $R_{11}(u) :: \neg P_1(u) \wedge P_{40}(u) \rightarrow$ 
 $status_u := CH; cl_u := id_v; dist_{(u,CH_u)} := 0; gn_u := id_u;$ 
 $R_{12}(u) :: \neg P_1(u) \wedge P_{41}(u) \rightarrow status_u := SN; cl_u :=$ 
 $id_v; dist_{(u,v)} := 1; gn_u := NeighCH_u;$ 
 $R_{13}(u) :: \neg P_1(u) \wedge P_{42}(u) \rightarrow status_u := SN; cl_u :=$ 
 $cl_v; dist_{(u,CH_u)} := dist_{(v,CH_v)} + 1; gn_u := NeighMax_u;$ 
 $R_{14}(u) :: \neg P_1(u) \wedge P_{43}(u) \rightarrow$ 
 $status_u := CH; cl_u := id_v; dist_{(u,CH_u)} := 0; gn_u := id_u;$ 
 $R_{15}(u) :: P_2(u) \wedge P_{44}(u) \rightarrow status_u := GN;$ 
 $R_{16}(u) :: P_1(u) \wedge P_{41}(u) \rightarrow status_u := SN; cl_v :=$ 
 $id_v; dist_{(u,v)} := 1; gn_u := NeighCH_u;$ 
 $R_{17}(u) :: P_1(u) \wedge P_{42}(u) \rightarrow status_u := SN; cl_u :=$ 
 $cl_v; dist_{(u,CH_u)} := dist_{(v,CH_v)} + 1; gn_u := NeighMax_u;$ 

/* Sending hello message */
 $R_0(u) :: hello(id_u, cl_u, status_u, dist_{(u,CH_u)});$ 
    
```

**A. Impact of density and network size on the stabilization time**

First, we study the impact of nodes degree and network size on the stabilization time. In figure 3(a), we have fixed a hops number  $k = 2$ . For each node degree of 3, 5 and 7 we consider a network size from 100 to 1000 nodes. Note that we generate  $d$ -regular graphs models using SNAP library, where  $d$  represents node's degree (number of neighbors for each node). For each given network size, we compute several series of simulations. We give the average stabilization time as the average of all values corresponding to simulation results. We note that the stabilization time increases as the number of nodes in the network increases. Furthermore, we note that

for arbitrary topologies, the average stabilization time is below  $n + 2$ , formal value proved in the worst case. Moreover, the number of transitions needed to reach a legal configuration appears stable when the network size increases (500 to 1000 nodes).

To observe the impact of the network density as illustrated in figure 3(b), we consider a network size of 100, 200 and 400 nodes and we vary the nodes degree. We observe that the stabilization time decreases as the nodes degree increases. The main reason is due to the fact that each node has more neighbors, thus during each transition, we have more nodes that fixed at the same time. With our approach, we have a better stabilization time with networks of high density.

**B. Scalability**

To examine the scalability of the proposed solution, we vary the number of nodes in the network at the same time as the density of connectivity. For  $k = 2$ , we consider a network size of 100 to 1000 nodes. For each value of the network size, we vary the density from 10% to 100%. Note that we generate Erdős-Renyi random graphs models using SNAP library. We obtain the 3D curve illustrated in figure 4. We note that except for low densities (10% and 20%), the stabilization time varies slightly with the increasing number of nodes. In case of a low network density, we observe a peak that is due to longer chains in the network topology. With these series of simulation, we can make two remarks. (i) The only determining factor with our approach is the density of connectivity and our solution is scalable. (ii) On average, for networks with an arbitrary topology, the stabilization time is far below that of the worst case ( $n + 2$  transitions).

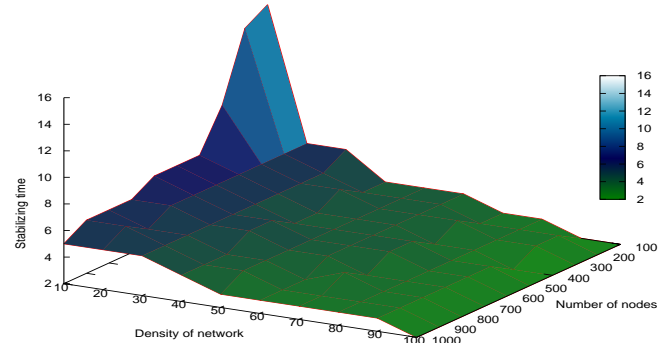


Fig. 4. Scalability

**C. Size and number of clusters**

As the density of connectivity is the determining factor for our algorithm, we evaluate the number of clusters obtained according to the network density. For  $k = 2$ , we consider a network size of 100, 500 and 1000 nodes. We vary the node degree from 5 to 100 neighbors. Figure 5(a) shows that regardless the number of nodes in network, we get less clusters when the number of neighbors increases. In fact, in denser



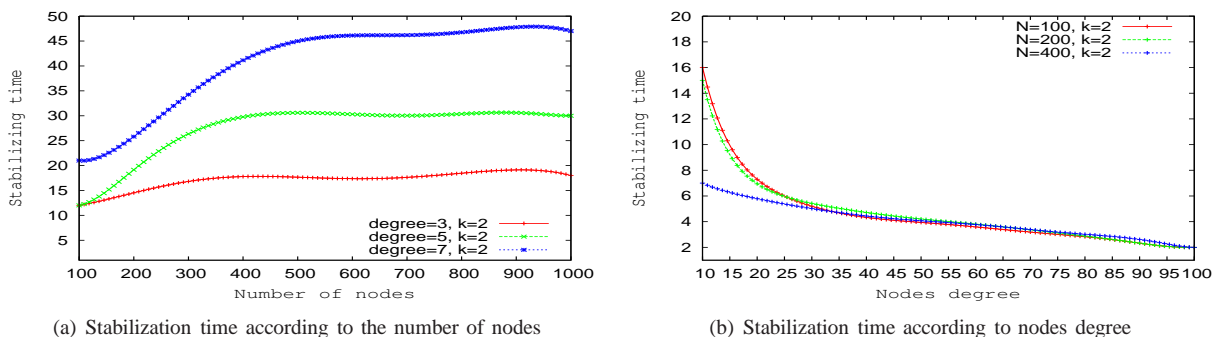


Fig. 3. Impact of nodes degree on the stabilization time

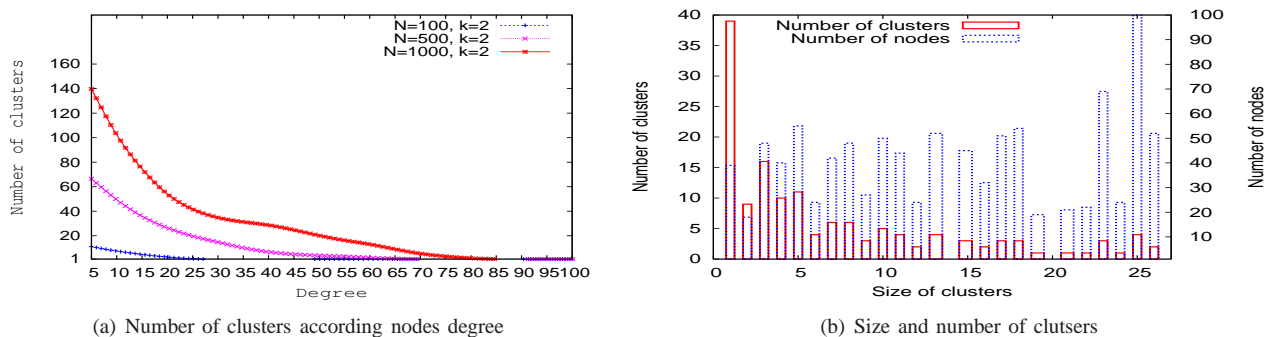


Fig. 5. Size and number of clusters

networks, nodes with the largest identity absorb more nodes into their clusters.

As we have more clusters with low density, we consider a network size of 1000 nodes with 5 neighbors for each node. We evaluate nodes distribution between clusters. We note that, as illustrates in figure 5(b), we have clusters of variable size. We have 39 singleton clusters, around 4% of the total number of nodes. We also note that the highest identity clusters include the more nodes its. The main reason is due to the fact that nodes choose as *CH* those with the highest identity.

D. Impact of the *k* parameter

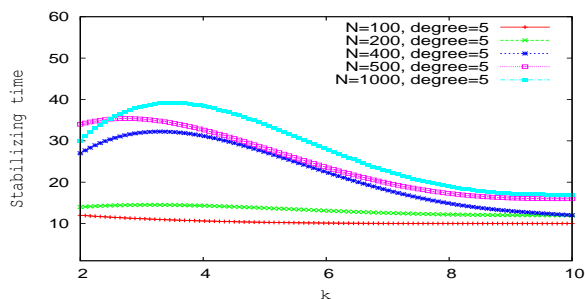


Fig. 6. Impact of *k* parameter

In order to observe the impact of the *k* parameter, we fix the node degree at 5 and we consider a network size of 100, 200,

400, 500 and 1000 nodes. For each network size, we vary the *k* parameter from 2 to 10. Figure 6 shows the stabilization time according to the variation of the *k* parameter. We observe that the stabilization time decreases as the *k* parameter increases. In fact, if *k* parameter increases and because the hello message contains the distance between each node *u* and its clusterhead, the sphere of influence of the largest nodes increase. Thus, nodes carrying fewer transitions to be fixed at a *CH*. In the end, we have fewer clusters. Nevertheless, in the case of a small value of the *k* parameter, we have more clusters with small diameters. Therefore, it requires more transitions to reach a stable state in all clusters. Note that regardless the value of the *k* parameter, the stabilization time is far below that of the worst case scenario ( $n + 2$  transitions).

E. Comparison with a well known solution based on message-passing model

In message-passing model, their exist few solutions. We compare our approach with a well known existing solution in message-passing model [8]. We compare these two approaches in terms of number of messages exchanged (*i.e* received messages) and number of clusters. We have implemented our algorithm and [8] on OMNeT++ environment simulation. Simulations are made within the same random graph.

In order to evaluate number of exchanged messages, we fix the node degree at 3 and 6 and we consider a network size from 100 to 1000 nodes. For each given network size, we compute several series of simulations. We give the average number of

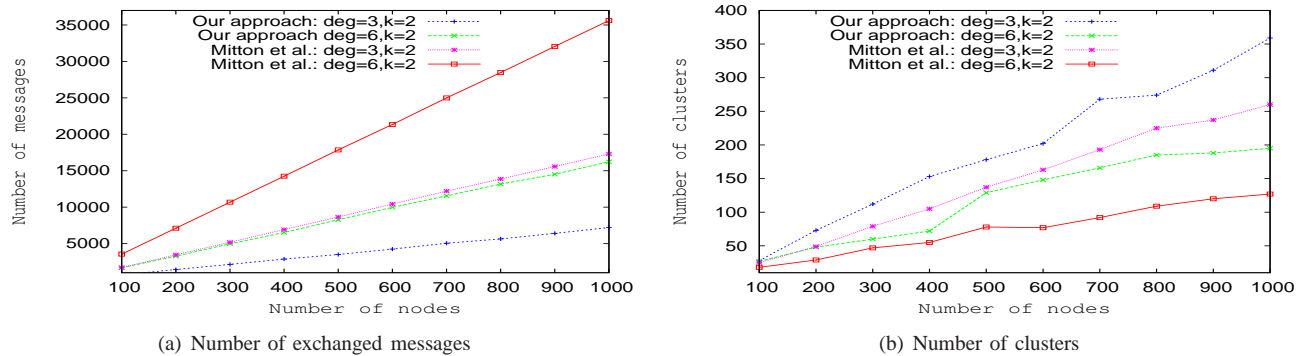


Fig. 7. Comparison with Mitton et al. [8]

exchanged messages in the network to achieve the legal state. Figure 7(a), show that our approach generates less messages than [8]. The main reason is due to the fact that our approach is based only on information from neighboring at distance 1 to build  $k$ -hops clusters. As opposed to solution [8], each node must know  $\{k+1\}$ -Neighboring, computes its  $k$ -density value and locally broadcasts it to all its  $k$ -neighbors. This is very expensive in terms of exchanged messages.

We also evaluate the number of clusters obtained by both approaches. As illustrates in figure 7(b), our approach built more than clusters. Moreover it generates less messages. This implies that our clusters are less densely. Therefore, easy to manage by the clusterhead and consumes fewer resources. Less densely clusters leads to a decrease communications and an optimization of resource consumption.

Note that we have made comparisons for  $k = 2$ . In fact, if  $k$  parameter increase, the broadcast area of solution [8] increases. Whereas our approach is based only on a neighborhood of a node at distance 1 to structure the network into non-overlapping  $k$ -hops clusters.

## VII. CONCLUSION

We presented a self-stabilizing asynchronous distributed algorithm based on a message-passing model. The proposed solution structures the network into non-overlapping clusters with diameter at most equal to  $2k$ . This structuring does not require any initialization. Furthermore, it is based only on information from neighboring nodes at distance 1. Contrary to other clustering algorithms, we have used an unique message to discover the neighborhood of a node at distance 1 and to structure the network into non-overlapping  $k$ -hops clusters.

Starting from an arbitrary configuration, the network converges to a legal configuration after at most  $n + 2$  transitions and requires at most  $n * \log(2n + k + 3)$  memory space. Experimental results show that for arbitrary topology networks, the stabilization time is far below the worst case scenario. A comparison with one of the best existing solution based on message-passing model show that we use fewer messages and stabilizing time is better.

As future work, we plan to implement mechanisms to balance clusters and maintaining the formed ones in case of

topology change.

## VIII. ACKNOWLEDGEMENTS

This work is supported in part by Regional Council of Champagne-Ardenne and European Regional Development Fund. The simulation are executed on Grid'5000 experimental testbed, hosted at the ROMEO HPC Center.

The authors wish to thank the reviewers and editors for their valuable suggestions and expert comments that help improve the contents of paper.

## REFERENCES

- [1] L. Blin, M. G. Potop-Butucaru, and S. Rovedakis, "Self-stabilizing minimum degree spanning tree within one from the optimal degree," JPDC, 2011, pp. 438 – 449.
- [2] A. Datta, L. Larmore, and P. Vemula, "Self-stabilizing leader election in optimal space," in SSS, 2008, pp. 109–123.
- [3] C. Johnen and L. Nguyen, "Self-stabilizing weight-based clustering algorithm for ad hoc sensor networks," in ALGOSENSORS, 2006, pp. 83–94.
- [4] N. Mitton, A. Busson, and E. Fleury, "Self-organization in large scale ad hoc networks," in MED-HOC-NET, 2004.
- [5] O. Flauzac, B. S. Hagggar, and F. Nolot, "Self-stabilizing clustering algorithm for ad hoc networks," ICWMC, 2009, pp. 24–29.
- [6] C. Johnen and L. Nguyen, "Self-stabilizing construction of bounded size clusters," ISPA, 2008, pp. 43–50.
- [7] C. Johnen and L. H. Nguyen, "Robust self-stabilizing weight-based clustering algorithm," TCS, 2009, pp. 581 – 594.
- [8] N. Mitton, E. Fleury, I. Guerin Lassous, and S. Tixeuil, "Self-stabilization in self-organized multihop wireless networks," in ICDCSW, 2005, pp. 909–915.
- [9] E. Caron, A. K. Datta, B. Depardon, and L. L. Larmore, "A self-stabilizing k-clustering algorithm for weighted graphs," JPDC, 2010, pp. 1159–1173.
- [10] E. W. Dijkstra, "Self-stabilizing systems in spite of distributed control," Commun. ACM, 1974, pp. 643–644.
- [11] A. K. Datta, S. Devismes, and L. L. Larmore, "A self-stabilizing  $O(n)$ -round  $k$ -clustering algorithm," in SRDS, 2009, pp. 147–155.
- [12] H. Attiya and J. Welch, Distributed computing: fundamentals, simulations, and advanced topics. John Wiley & Sons, 2004.
- [13] G. Tel, Introduction to Distributed Algorithms. Cambridge University Press, 2000.
- [14] M. Ba, O. Flauzac, B. S. Hagggar, F. Nolot, and I. Niang, "Self-stabilizing  $k$ -hops clustering algorithm for wireless ad hoc networks," in ACM IMCOM, 2013.
- [15] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in Simutool, 2008, pp. 60:1–60:10.
- [16] SNAP: Stanford Network Analysis Platform. <http://snap.stanford.edu>
- [17] F. Cappello et al., "Grid'5000: A large scale and highly reconfigurable grid experimental testbed," in GRID, 2005, pp. 99–106.

# Goal Processing and Semantic Matchmaking in Opportunistic Activity and Context Recognition Systems

Gerold Hoelzl, Marc Kurz, Alois Ferscha  
 Institute for Pervasive Computing  
 Johannes Kepler University Linz  
 Linz, Austria  
 surname@pervasive.jku.at

**Abstract**—The **Opportunistic Sensing Paradigm** shifts away from the configuration of activity- and context recognition systems from design time of the system to a dynamic runtime configuration. Systems following this new paradigm have to autonomously configure themselves during runtime according to a stated recognition goal and the available sensing infrastructure. A crucial point therefore is to find a methodology to express the recognition goal itself and to map the requirements specified by the stated recognition goal to the capabilities of the available sensors in the ecosystem of the users. This paper presents an approach for a semantic engine capable of processing and matching a given recognition goal to the semantically described sensing infrastructure. This matching allows to autonomously spot and quantify the best set of sensors available at any point in time that can contribute to the stated recognition goal and therefore being configured to a sensing ensemble.

**Index Terms**—Goal Processing; Sensor Networks; Activity and Context Recognition; Opportunistic Sensing

## I. INTRODUCTION

Sensor networks enable the sensing of the physical world and detect context and activity, which is a key to build and develop intelligent environments [1]. Different Frameworks exist that handle inferring context and activity out of raw sensor data. Their common disadvantage is that the used sensing infrastructure and the context that has to be recognized must be defined at the design time of the system [2]. A dynamic change in the sensing infrastructure in terms of the spontaneous availability of new sensors or the change of the recognition purpose respectively the recognition goal [3] is not supported by these systems. Due to the emerging plethora of sensing devices (e.g., smart phones or smart watches) with integrated sensing capabilities in the ecosystem surrounding the user, the necessity arises to federate these devices in an autonomous, adaptive and goal oriented manner to relieve developers from low level, platform specific concerns. Kurz et al. present in [3] methodologies to semantically describe the recognition capabilities of sensors in terms of labels reflecting meaning in the real world (e.g., *Walk* or *Drinking-Coffee*). They propose *ExperienceItems* that encapsulate the necessary stages of the *Activity Recognition Chain* [4] to infer activity and context information out of the raw sensor data readings. In

addition, a metric, the *Degree of Fulfillment (DoF)* is defined, that quantifies how good a sensor can be used to detect a specific label (i.e., activity class) as a value between  $[0, 1]$ . Furthermore, a way to abstract from the low level access details yielding to a common accessible interface for different sensing devices using *Sensor Abstractions* is described and evaluated. Using these presented methodologies, we can see a sensor as a label delivering entity that can be queried for its capabilities, and the needed machine learning techniques can be instantiated during runtime to infer activity and context information. To direct the autonomous planning process [5] in selecting the best suitable sensing devices, we propose a goal oriented approach that defines at a high semantic level how the system should behave. Goal oriented approaches have proven their flexibility and effectiveness in various research domains like *Operations Research* [6], *Requirements Engineering* [7] and *Mobile Agent Systems* [8]. The common objective is that a goal specifies at an abstract level what the system should achieve but not how. Multiple paths exist at each point in time on how the goal can be achieved. In this work we adapt the goal methodology and use a goal oriented approach to direct the configuration of an activity and context recognition system. This is a novel approach as the sensing infrastructure needs not to be defined at design time of the system and can change even during its runtime. It is autonomously configured and adapted according to the stated recognition goal. In detail, the hypotheses valid within this paper can be formulated as follows:

- (H) *The explicit formulation of a recognition goal is a suitable way of directing the autonomous configuration of an activity and context recognition system.*

The remainder of the paper is structured as follows: Section II proposes a methodology on how activity can be understood and modeled. This is a crucial point to define the necessary terms a recognition goal can be composed of. Based upon our activity modeling approach that encapsulates our understanding of what an activity is, Section III discusses a technique on how to formulate a recognition goal based on this understanding. Section IV shows how a formulated

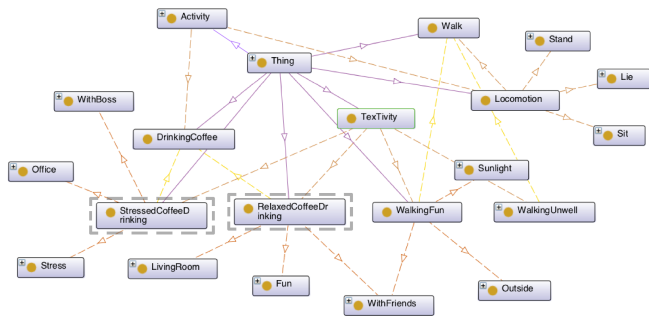


Fig. 1: Partial view of the approximately 200 modeled *Context* and *Activity Relations* gathered during a large scale kitchen experiment presented in [11] using the Protege-OWL modeling tool. Activities and Context are modeled using the relations  $\mathcal{R}_{isRelated}$ ,  $\mathcal{R}_{Combination}$ , and  $\mathcal{R}_{has\_A}$ .

recognition goal is processed, refined, semantically reasoned and quantified, and finally mapped to the available sensing ecosystem. Section V evaluates the proposed approach in a real-time setting and Section VI closes with a discussion of the hypotheses and a conclusion.

## II. ACTIVITY RELATIONS MODELING

To formulate a goal that describes the recognition purpose of the system at an abstract, semantic level, the necessity evolves to understand, define, model, and relate *Activities* and *Context* that can be stated to the system. We see and define *Activities* as a much more complex thing rather than a simple "sequence of actions performed" [9] or "hierarchies of activities of daily living" [10]. Its meaning is not only defined by the performed physical activity itself. It is defined in combination with a situation respectively the context. To model these relations we use an *Ontology* that builds the semantic knowledge base and represent the *Activity Relations* gathered during a large scale kitchen experiment presented in [11].

In each context, a physical activity can have a complete different meaning. From our point of view *Activity* can be seen as any information that characterizes the behavior of an individual interweaved with additional context [12]. We model these relations using the *TextTivity*-Concept (as shown in (1)). This concept allows to relate activities to different contexts and assign different meaning to these relations. *TextTivity* therefore is an amalgam of *context* and *activity* highlighting the close relation of both concepts. Fig. 1 shows a partial view of the modeled relations. Two *TextTivities* namely *StressedCoffeeDrinking*, and *RelaxedCoffeeDrinking* are modeled. Both *TextTivities* are related to same physical activity *DrinkingCoffee* but assign a different meaning of this activity in relation to the two contexts [*WithBoss*, *Office*, and *Stress*]→*StressedCoffeeDrinking* and [*WithFriends*, *Outside*, and *Fun*]→*RelaxedCoffeeDrinking*. According to the definition of *TextTivities* we named the Ontology to model these relations *TextTivity-Ontology*. Using the definition of *TextTivities* (as defined in (1)) allows us to link a set of context literals

(without activity) [ $\mathcal{R}_{Combination}$ ] to a set of activity literals [ $\mathcal{R}_{isRelated}$ ] and give this relation a "meaning". Furthermore, by knowing the relations of activities [ $\mathcal{R}_{has\_A}$ ], the substitution and reasoning of activities is possible by combining activities at different levels of abstraction. This is especially useful if no sensor can be found in the surrounding of the user that is explicitly designed to detect the stated recognition goal. Using the modeled relations of activities, we can semantically reason which activities the stated recognition goal is composed of, and use sensors that can detect these semantically related activities instead. This allows the dynamic configuration of sensing ensembles and exploits the full flexibility of the opportunistic sensing approach, as the recognition goal can be processed, reasoned and matching methodologies can be applied to substitute activities out of related ones. How this is done is described in detail in Sections III and IV.

$$TextTivity \rightarrow \{Activity\} \odot \{Context \setminus Activity\} \quad (1)$$

We see *Activities* as a multidimensional labyrinth of complex actions. They can happen interleaved, concurrent or in any other temporal or causal relationship. Making it even more complex, *activities* can happen in combination with different contexts that influence their meaning. Consider the activity "Talking" either in the context of watching a football match or improperly in a class room. Both activities "Talking" refer to the same "physical" action but with a complete different meaning in the particular context.

## III. RECOGNITION GOAL FORMULATION

One of the crucial things when developing an opportunistic activity and context recognition system that adapts autonomously during runtime is to find a way to formulate the recognition goal explicitly to relieve developers from low level, platform specific concerns while federating the sensing infrastructure. The explicitly stated recognition goal has to be reason- and processable by the opportunistic system. This enables the autonomous selection of the best set of available sensors, called *ensemble* [3], according to the stated recognition goal. We see a sensor as a label delivering entity at a semantic level reflecting meaning in the real world that can be used and queried for its capabilities, can be quantified during runtime using the *Degree of Fulfillment (DoF)* metric, and can be dynamically instantiated at any point in time. How the sensor ensemble is configured and the corresponding activity recognition chains are instantiated by using sensor abstractions, sensor-self-descriptions, and experience-items is described in [3].

To formulate the recognition goal we use a derivation of the *Context Predicates* presented in [13]. There, Stevenson and Dobson present Context Predicates to reason information in smart environments (e.g., smart homes). The Context Predicates are used as entry points into the modeled domain knowledge captured in form of an Ontology. Furthermore, they are used to reason about the modeled knowledge. If for example, one wants to know if Bob is at home, and the information that Bob is in the living room is given (by

a particular sensor), the domain knowledge can be used to reason that the living room is located in the house and to infer that Bob is at home. The reasoning capabilities can be used to infer and refine knowledge that is not explicitly given by a particular sensor. Even if we do not have the explicit information that Bob is at home (e.g., no sensor is available that can deliver this information explicitly), we can infer this knowledge by using other sensors that detect that Bob is in the living room. Having the semantic modeled relation of *Home*  $\leftrightarrow$  *Living Room* we can infer that Bob is at home without having the explicit knowledge of a particular sensor. The syntax of the *Context Predicate* is defined in (2).

$$\langle CP \rangle = \langle \text{subject}, \text{predicate}, \text{object} \rangle \quad (2)$$

The following examples clarify the use of Context Predicates. A Context Predicate can be for example  $\langle \{Bob\}, \text{located}, \text{livingRoom} \rangle$ , querying if Bob is located in the living room. This query can be evaluated to *true* or *false*. Another example is  $\langle ?, \text{located}, \text{livingRoom} \rangle$ . Using the question mark, querying for all subjects that are located in the living room is possible. The last example  $\langle \text{Person}, ?, \text{livingRoom} \rangle$  queries for all predicates (activities) persons are performing in the living room.

Following the approach of using Context Predicates, that showed to be a promising solution as a goal description using domain knowledge and reasoning, we enhanced them to more closely fit our proposed *TextTivity*-Ontology to enable a quick and precise formulation of the recognition purpose of the system. As we talk about activity and context recognition, we renamed the *predicate* into *Activity* and made the term *object* more generic in calling it *Context*. Furthermore we added the entity *TextTivity* to allow its direct statement as a goal. We permit to state each item in form of a set to allow multiple objectives per item being stated in a single goal and define a *TextTivity Predicate* as follows (3):

$$\langle TP \rangle = \langle \{Subject\}, \{Activity\}, \{Context\}, \{TextTivity\}, \{Constraint\} \rangle \quad (3)$$

The definition of a *TextTivity Predicate* can be used to formulate a goal like  $\langle \{Bob, Paul\}, ?, \{LivingRoom, Bathroom\}, ?, \{ \} \rangle$ . The formulated goal configures a system that detects all activities and *TextTivities* (as indicated by the question marks) of Bob and Paul in the living- and bathroom. As from the systems point of view, the best ensemble is always defined by having the highest recognition accuracy concerning the stated context and activity items, the necessity may evolve to further optimize such an ensemble in terms of energy consumption, size and weight, infrastructure or body-worn sensors, or the maximum numbers of sensors that are configured to an ensemble. As these constraints cannot be foreseen now, the *TextTivity predicates* allow to state them in an open and extendable way. In the previous example, no constraints were defined. So the best available sensor ensemble is configured regardless its quantitative properties. If one wants to formulate the same

recognition goal, but assuring that the configured ensemble is above a stated recognition accuracy, e.g., with a *Degree of Fulfillment (DoF)* above 89%, it has to be added as constraint ( $\langle \{Bob, Paul\}, ?, \{LivingRoom, Bathroom\}, ?, \{[DoF] : 0.89\} \rangle$ ) instructing the system to configure a sensing ensemble with a DoF above 89% if possible, or fail otherwise. Formulating subjects, activities, context, *TextTivities* and constraints in form of sets is an elegant and fast way to state the recognition purpose of the system for multiple objectives.

Activity and Context is not limited to one, synchronized stream of information that can be described and formulated using one single *TextTivity predicate*. To meet these concerns, we identified two ways of combining *TextTivity Predicates* for activity and context recognition systems: (i) a logical combination and (ii) a temporal combination. The logical combination can be done e.g., with an *AND* or *OR* connector, allowing to state multiple recognition purposes to the system that are handled in parallel. An example is shown in (4) where the logical combination *AND* is used to combine two *TextTivity Predicates*.

$$\begin{aligned} &\langle \{Bob\}, ?, \{LivingRoom\}, ?, \{ \} \rangle \text{ AND} \\ &\langle \{Paul\}, ?, \{Bathroom\}, ?, \{ \} \rangle \end{aligned} \quad (4)$$

The purpose of the recognition goal in (4) is to detect all Activities and *TextTivities* that are performed by Bob in the livingroom *AND* all Activities and *TextTivities* that are performed by Paul in the bathroom. Therefore, two *TextTivity Predicates* are defined that formulate the recognition goals. These two *TextTivity Predicates* are then combined using the logic *AND* operation to get the overall recognition goal for the activity and context recognition system.

Allen defined in [14] temporal operators like *Overlaps*, *Meets*, *Equal*, *Before*, *During*, *Starts* and *Finishes*. By using this set of operators the whole space of possible temporal relationships can be represented. Using these operators to combine *TextTivity Predicates* allows to formulate timing behavior between multiple *TextTivity Predicates*. We think of recognition goals as shown in (5) and (6). In (5) the purpose of the recognition goal is to detect that Bob brushed his teeth *Before* he was having breakfast that could be an interesting context information for an child education system. In (6) the recognition goal is the detection of Paul reading his emails on Apple's iPad while having lunch in the office, which might also be useful information for systems helping to avoid the burn-out syndrome.

$$\begin{aligned} &\langle \{Bob\}, \{BrushTeeth\}, \{ \}, \{ \}, \{ \} \rangle \text{ BEFORE} \\ &\langle \{Bob\}, \{Breakfast\}, \{ \}, \{ \}, \{ \} \rangle \end{aligned} \quad (5)$$

$$\begin{aligned} &\langle \{Paul\}, \{ReadingMail\}, \{IPad\}, \{ \}, \{ \} \rangle \\ &\text{ DURING} \\ &\langle \{Paul\}, \{ \}, \{ \}, \{OfficeLunch\}, \{ \} \rangle \end{aligned} \quad (6)$$

TABLE I: SIMILARITY SCORES OF ONTOLOGICAL RELATIONS TO MATCH ADVERTISEMENTS OF RESOURCES (E.G., SENSOR) TO REQUESTS [15].

Similarity-Score( $C_R, C_A$ )	Taxonomic-Ontological Relation	Computation
1.0	$C_A \equiv C_R$	1.0
1.0	$C_A \subseteq C_R$	1.0
[0,1]	$C_R \subseteq C_A$	$\frac{ A(C_A) }{ A(C_R) }$
[0,1]	$\neg(C_R \cap C_A \subseteq \perp)$	$\frac{ A(C_A) \cap A(C_R) }{ A(C_R) }$
0.0	$(C_R \cap C_A \subseteq \perp)$	0.0

The use of *TexTivity*-Predicates to define the recognition purpose of an activity and context recognition system in form of an semantic abstracted, high level goal, is a flexible way to direct the autonomous and dynamic configuration of such a system. Having the option of combining multiple *TexTivity* Predicates either with logical or temporal operators makes this approach even more powerful and allows to represent the nature of activities more clearly.

#### IV. SEMANTIC ENGINE

Using the semantic information modeled in the *TexTivity*-Ontology is the key aspect to utilize the full power and flexibility of the opportunistic approach. To configure sensor ensembles according to the formulated high level recognition goal, we use the semantic information modeled in the *TexTivity* Ontology. Using the modeled *Context*, *Activity* and *TexTivity* relations of a domain allow to reason which Context- and Activity attributes can be substituted by related ones. As the *TexTivity* Ontology models the concepts and relations of a domain, each possible sensor output in terms of labels is an entry point in the modeled domain knowledge. The labels that can be delivered by sensors are the connection point between the "semantic"-sensors, the *recognition goal* and the *TexTivity* Ontology. This allows the combination of sensors according to the defined relations in the *TexTivity* Ontology and to reason knowledge at a semantic level. If there is, for example, no sensor in the environment that can be explicitly used to detect a stated recognition goal, we can search for sensors that can be used instead and therefore substituting the not available sensor entity. According to the defined semantic relations in the *TexTivity* Ontology, we can reason how the formulated recognition goal can be substituted by related contextual information. In [15] Bandara et al. propose a way based on the semantic similarity concepts presented by Lin [16] on how to match the advertisements of a resource (e.g., a (self-described) sensor) to a request (e.g., the recognition goal respectively the recognition purpose). The matching scores between the concepts are shown in Table I.

To find the best sensor ensemble according to the stated recognition goal, we define a *Goal Function* (as shown in (7)) that uses the semantic similarity concepts presented in [16] and matches the requested capabilities of the recognition goal to the advertised recognition capabilities of the sensors

(as described conceptual in Table I). The function weights the resulting *DoF* according to the number of labels (and their single DoFs) that can be satisfied by the sensor ensemble and returns a value between [0,1] that reaches a maximum for the best set of sensors.

$$DoF = \frac{1}{LtR} \sum_{i=0}^{n-1} DoF_i \quad (7)$$

$LtR$ =number of *Labels* the sensor is queried for

$n$ =number of *Labels* satisfied by the sensor ( $n \leq LtR$ )

$DoF_i$ =DoF of a single label with index  $i$

In case where no sensor is found in the ecosystem of the user that can explicitly be used to detect the stated recognition goal, *Goal Substitution* and *Reasoning* using the captured *Domain Knowledge Relations* takes place. By knowing the semantic relations of context and activities, we can reason which sub-context can be used to reason context at a higher semantic level. This is an extremely powerful approach as we can use sensors that were not explicitly designed to detect the stated recognition purpose. An example is the recognition goal  $\langle \{\}, \{Locomotion\}, \{\}, \{\}, \{\} \rangle$  that formulates to detect *Locomotion*. If we can not find a sensor in the environment, that can explicitly be used to detect *Locomotion* we use the semantic information modeled in the *TexTivity* Ontology. We refine *Locomotion* into its related sub-activities ( $\mathcal{R}_{has\_A}$ )  $\rightarrow$  *Walk*, *Stand*, *Lie* and *Sit* as shown in Fig. 1 and search for sensors that are capable of recognizing these activities. We rank the found sensors according to the *Goal Function* defined in (7) and select the highest ranked as ensemble. The example highlights, the use of sensors to detect *Walk*, *Stand*, *Lie* and *Sit* that were not designed to detect *Locomotion* explicitly. Only because the *semantic relations* of *Locomotion* to these four activities are known, and modeled in the *TexTivity* Ontology, we can combine them to detect *Locomotion*.

As the reasoning capabilities are not limited to one refinement step  $\{A \rightarrow \{B, C\}\}$ , the refinement of activities and context may involve the recursive evaluation of an entire tree of related context  $\{A \rightarrow \{B, C\}; B \rightarrow \{U, V, W\}, C \rightarrow \{X, Y\}\} \rightarrow \{\dots\}$ . This can take as many steps as necessary to decide if the formulated recognition purpose is satisfiable or not. Furthermore, reasoning of a recognition goal, expressed in form of a *TexTivity* Predicate, is a reasoning in multiple dimensions as it can be done for all elements contained in the *TexTivity* Predicate as exemplarily shown in Fig. 2. The two presented methodologies, (i) the *semantic matchmaking* of the recognition goal expressed using *TexTivity* Predicates to the sensing infrastructure and (ii) the *goal splitting, substitution* and *quantification* according to the defined *Goal Function* (7) are implemented in a *Semantic Engine* that handles the *Semantic Concept Matching* between the sensing ecosystem and the stated recognition goal as shown in Figure 3. In the following Section V we evaluate the presented methodologies using a reference implementation of an opportunistic activity and context recognition system called the OPPORTUNITY Framework [3].

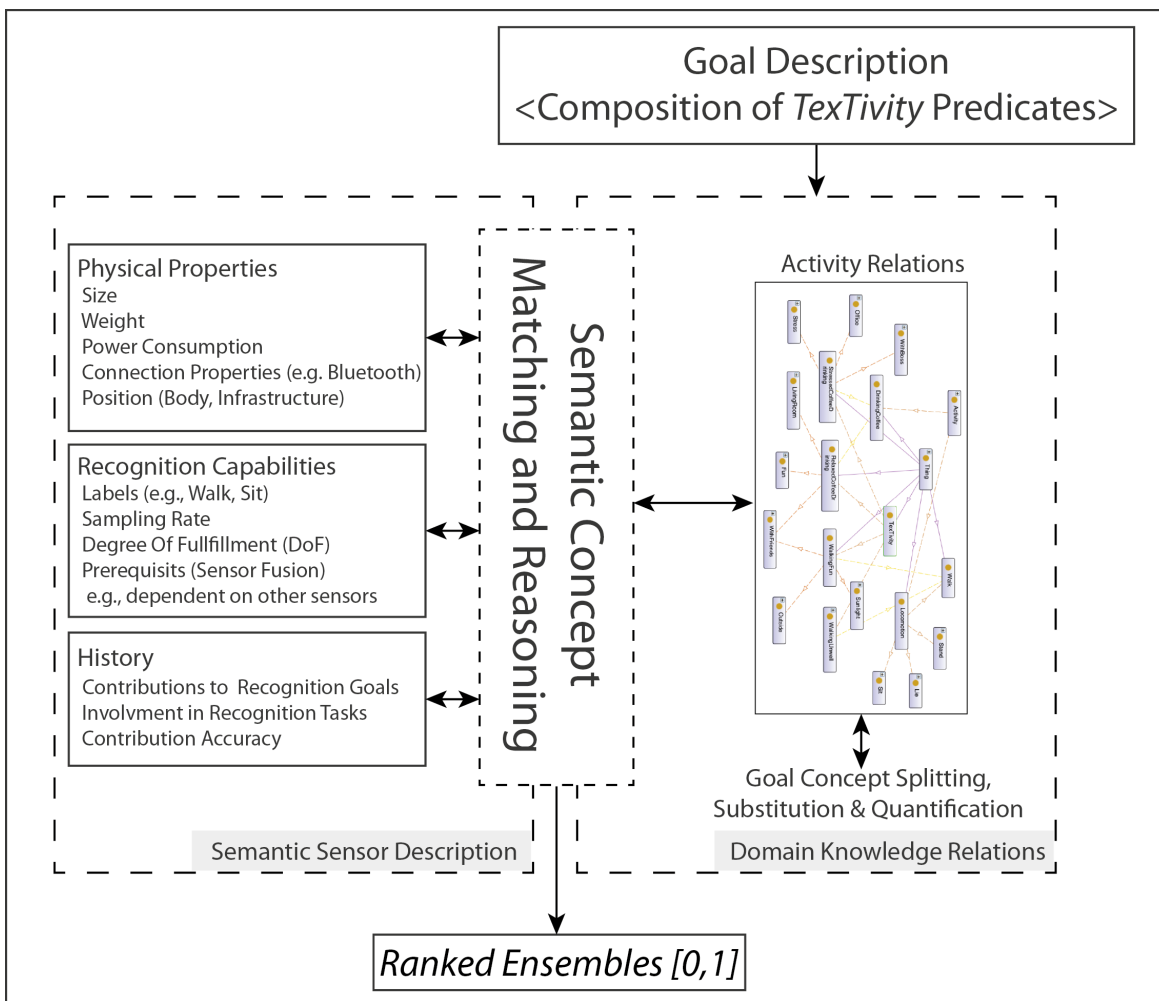


Fig. 3: Semantic Engine used for Processing the stated Recognition Goal in mapping, reasoning and substituting it to the available sensors in the ecosystem of the user.

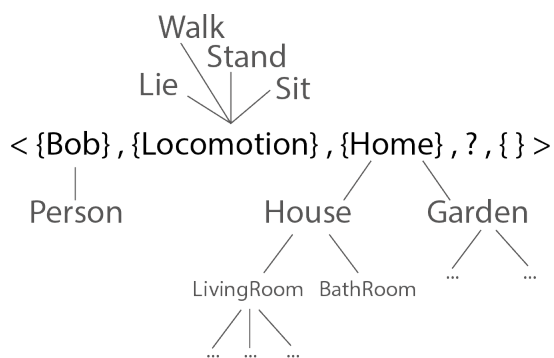


Fig. 2: Reasoning of TextTivity Predicates in multiple dimensions for the recognition goal  $\langle \{Bob\}, \{Locomotion\}, \{Home\}, ?, \{ \} \rangle$  according to the defined relations in the TextTivity Ontology.

### V. EVALUATION

To evaluate the approach of using high level recognition goals to direct the dynamic configuration of an activity and

context aware system, we developed the OPPORTUNITY Framework [3]. The framework is a reference implementation of an opportunistic context and activity recognition system. It allows to state high level recognition goals during runtime in form of TextTivity predicates that direct the dynamic and autonomous configuration of the system. It uses the goal processing, reasoning and semantic matchmaking capabilities of the *Semantic Engine* described in Sections III and IV to configure sensor ensembles to fulfill the stated high level recognition goal. A similar setup was already successfully used by our group to test the configuration of sensor ensembles [17]. For the evaluation we set up a real time scenario with body worn sensors to test and evaluate the presented approach. We picked a rather easy activity set for evaluating, as the goal is not to work with highly sophisticated and complex activity classes. The handling of complex activities using dynamically configured Hidden Markov Models with a similar sensor setup is presented in [17]. For testing we used the activity classes of the *modes of locomotion*  $\rightarrow$  (Walk, Sit, Stand, Lie). We used four XSense-Mti-sensors that deliver

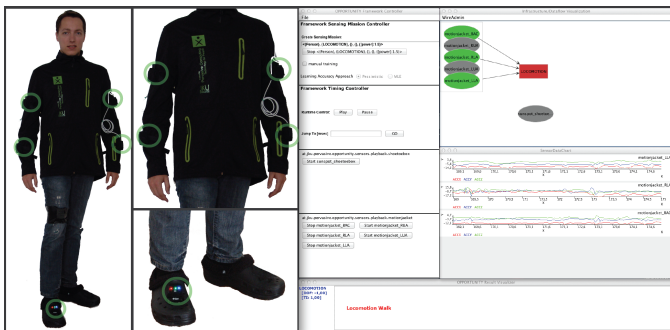


Fig. 4: Experimental setup of the on-body sensors to test and evaluate the autonomous configuration of an Activity and Context recognition system in a realtime setting.

triaxial acceleration data. The sensors were mounted on the left and right upper-/lower arm and are named according to their position *motionjacket\_RUA* (right-upper-arm), *motionjacket\_RLA* (right-lower-arm), *motionjacket\_LUA* (left-upper-arm) and *motionjacket\_LLA* (left-lower-arm). Additionally, one SunSPOT accelerometer sensor was attached to the right shoe (*sunspot\_shoetoebox*). Fig. 4 shows the sensor setup and the corresponding realtime schematic of the runtime configured sensor ensembles using the OPPORTUNITY Framework. The four *motionjacket* sensors were trained to detect *Walk*, *Sit*, *Stand*, and *Lie* using the OPPORTUNITY Dataset [11]. The *sunspot\_shoetoebox* was trained to detect *Locomotion*. In the following two scenarios the *TexTivity* predicate  $\langle \{ \}, \{ \text{Locomotion} \}, \{ \}, \{ \}, \{ \} \rangle$ , formulating the *recognition goal* to detect the activity *Locomotion*, was stated to the system. In the first scenario, all sensors were available (Fig. 5a) as indicated by the green bubbles. The OPPORTUNITY Framework queried the available sensors for their recognition capabilities. One sensor, the *sunspot\_shoetoebox*, was found that can explicitly be used to recognize *Locomotion*. The goal refinement process stopped as a sensor was found that is explicitly dedicated to detect *Locomotion*. So this sensor is selected as ensemble (as indicated by the black arrow), the corresponding recognition chain is dynamically instantiated, and the recognition process is initiated.

In the second scenario, we turned off the *sunspot\_shoetoebox* sensor as indicated by the grey bubble (Fig. 5b). The four *motionjacket* sensors were still available. In this case, no sensor is available that can explicitly be used to detect *Locomotion*, as the *motionjacket* sensors are trained to detect *Walk*, *Sit*, *Stand* and *Lie*. Again, the OPPORTUNITY Framework queried the available sensors for their recognition capabilities. As no sensors were found by the framework that can explicitly be used to detect *Locomotion*, the *Semantic Engine* initialized the *Goal Refinement* process according to the semantically modeled relations of the *TexTivity Ontology*. The OPPORTUNITY Framework autonomously reasoned the related activities of *Locomotion* ( $\mathcal{R}_{has\_A} \rightarrow \{ \text{Walk}, \text{Stand}, \text{Lie}, \text{Sit} \}$ ) (Fig. 1). Knowing the related activities, the framework queried for

TABLE II: SEMANTIC SCORES FOR THE SEMANTICALLY REASONED RECOGNITION GOAL “LOCOMOTION” ACCORDING TO THE GOAL FUNCTION (7).

Sensor	Walk	Stand	Lie	Sit	Score
motionjacket_RUA	0.700	0.770	0.980	0.860	0.828
motionjacket_RLA	0.690	0.380	0.900	0.950	0.730
motionjacket_LUA	0.780	0.670	0.970	0.930	0.838
motionjacket_LLA	0.790	0.320	0.480	0.730	0.580

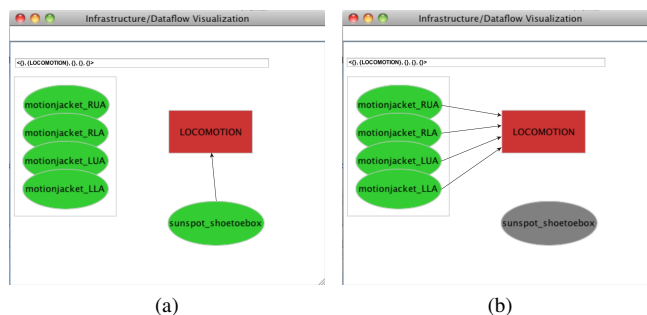


Fig. 5: Visual, real-time representation of the physical sensors used in the scenario to evaluate the *Goal Description and Semantic Matchmaking* in a varying sensor ecosystem.

sensors that can detect the related activities. Four sensors, the *motionjacket\_RUA*, *motionjacket\_RLA*, *motionjacket\_LUA* and *motionjacket\_LLA* were found that can be used. The *Goal Function* (7) was evaluated for each sensor to quantify its capabilities according to the reasoned recognition goal as shown in Table II. After evaluating the *Goal Function*, the four *motionjacket*-sensors were configured to an ensemble as the *Condorcet’s jury theorem* [18] states that if the probability of a single vote is greater than 0.5 (*DoF/Score*), then the probability that the majority decision is correct is increased. The corresponding recognition chains of the single sensors are dynamically instantiated and are fused using *majority voting* (Fig. 5b). Out of the four delivered labels (*Walk*, *Stand*, *Lie*, *Sit*) we can reason their relation on a higher semantic level inferring *Locomotion* using the modeled activity relations in the *TexTivity Ontology* that are utilized by the *Semantic Engine*. The evaluation scenario showed that it is possible to combine sensors according to semantically modeled information in a realtime system. We showed that the system can autonomously configure and adapt an activity and context recognition system dependent on the available sensing infrastructure and the stated recognition goal during runtime. Even if no sensors are available to detect the stated recognition goal explicitly, the system autonomously utilizes the semantically modeled information in the *Semantic Engine* in two ways. First, in a top-down fashion, it queries for related activity and context according to the stated *Recognition Goal* and then it quantifies sensors using a *Goal Function* (as shown in (7)) that can be used to detect them (e.g.,  $\text{Locomotion} \rightarrow \{ \text{Walk}, \text{Stand}, \text{Lie}, \text{Sit} \}$ ). Second,



the system uses the semantically modeled information in a bottom-up fashion, to reason context at a higher semantic level out of the semantic data delivered by the sensors (e.g., {Walk, Stand, Lie, Sit}→Locomotion).

## VI. CONCLUSION

Sensors providing information to detect human activities are embedded in more and more artifacts of everyday living (e.g., smartphones, watches, cameras, clothing, etc.). Using these artifacts motivates the shift away from deploying "application specific" sensors and to support developers in relieving them from low level, platform specific concerns. We use sensors that just happen to be available and utilize them in an goal oriented, opportunistic way for activity and context recognition. We can handle and exploit heterogeneous resources using *Sensor Abstractions* and dynamically instantiate the necessary stages of the *Activity Recognition Chain* to infer activity and context information out of the raw sensor data readings using *ExperienceItems*. We see each sensor as a label delivering entity that can be queried for its capabilities.

As central contribution we propose a *Semantic Engine* that matches the stated recognition goal in form of *TextTivity Predicates* to the available, semantically described sensor ecosystem. Therefore it utilizes modeled *context* and *activity relations* in the *TextTivity Ontology* that can be reasoned and inferred to combine sensors at different semantic levels. To direct the autonomous and adaptive configuration, we use a goal-oriented approach. The goal encapsulates a high level directive that defines, at a semantic level, what the system should do and how it should behave. As syntactical formalism, to describe and formulate the recognition goal we propose the use of *TextTivity-Predicates* as a precise, flexible, and reasonable way of formulating the recognition purpose of the Activity and Context recognition system.

The autonomous and adaptable combination and use of sensors at a semantic level provide the possibility to use sensors that are already out there in the environment and eliminates the need to deploy more and more application specific sensors. The fact that the system can react on changes in the sensing infrastructure (e.g., due to sensor failures) increases its stability and robustness compared to traditional systems. An essential point for the scalability of the opportunistic sensing approach is the fact that sensors, processing, and communication resources are only used and configured to ensembles, if they are needed to fulfill the stated recognition goal.

The evaluation clearly highlights that the research hypotheses holds that a goal oriented, top-down configuration approach for autonomously adapting an activity and context recognition system during runtime offers a high flexibility in terms of combining sensors at a semantical level resulting in a stable, and accurate activity and context recognition system envisioning the autonomous and dynamic creation of sensing infrastructures of massive scale during runtime.

## ACKNOWLEDGMENT

The project OPPORTUNITY acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 225938.

The project PowerIT acknowledges the financial support of the FFG FIT-IT under grant number: 830.605.

## REFERENCES

- [1] L. Benini, E. Farella, and C. Guiducci, "Wireless sensor networks: Enabling technology for ambient intelligence," *Microelectron. J.*, vol. 37, no. 12, pp. 1639–1649, 2006.
- [2] D. Roggen *et al.*, "Opportunity: Towards opportunistic activity and context recognition systems," in *Proceedings of the 3rd IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications (AOC 2009)*. Kos, Greece: IEEE CS Press, June 2009, pp. 1–6.
- [3] M. Kurz *et al.*, "The opportunity framework and data processing ecosystem for opportunistic activity and context recognition," *International Journal of Sensors, Wireless Communications and Control, Special Issue on Autonomic and Opportunistic Communications*, pp. 102–125, December 2011.
- [4] D. Roggen, K. Förster, A. Calatroni, and G. Tröster, "The adarc pattern analysis architecture for adaptive human activity recognition systems," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–18, 2011.
- [5] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.
- [6] T. Ellinger, G. Beuermann, and R. Leisten, *Operations Research*. Springer, 2003.
- [7] A. Van Lamsweerde, "Goal-oriented requirements engineering: A guided tour," in *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering*, ser. RE '01. Washington, DC, USA: IEEE Computer Society, 2001, pp. 249–263.
- [8] A. S. Rao and M. P. Georgeff, "BDI-agents: from theory to practice," in *Proceedings of the First Intl. Conf. on Multiagent Systems*, San Francisco, 1995, pp. 312–319.
- [9] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine Recognition of Human Activities: A Survey," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 11, pp. 1473–1488, Sep. 2008.
- [10] X. Hong, C. Nugent, M. Mulvenna, S. McClean, B. Scotney, and S. Devlin, "Evidential fusion of sensor data for activity recognition in smart homes," *Pervasive Mob. Comput.*, vol. 5, pp. 236–252, June 2009.
- [11] D. Roggen *et al.*, "Collecting complex activity data sets in highly rich networked sensor environments," in *Proceedings of the Seventh International Conference on Networked Sensing Systems (INSS), Kassel, Germany*. IEEE Computer Society Press, June 2010, pp. 233–240.
- [12] A. K. Dey, "Understanding and using context," *Personal and Ubiquitous Computing*, vol. 5, pp. 4–7, 2001.
- [13] J. Ye, G. Stevenson, and S. Dobson, "A top-level ontology for smart environments," *Pervasive Mob. Comput.*, vol. 7, pp. 359–378, June 2011.
- [14] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, pp. 832–843, Nov. 1983.
- [15] A. Bandara, T. Payne, D. De Roure, N. Gibbins, and T. Lewis, "A pragmatic approach for the semantic description and matching of pervasive resources," in *Proceedings of the 3rd international conference on Advances in grid and pervasive computing*, ser. GPC'08, Springer Berlin, Heidelberg, 2008, pp. 434–446.
- [16] D. Lin, "An information-theoretic definition of similarity," in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML '98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 296–304.
- [17] G. Hölzl, M. Kurz, and A. Ferscha, "Goal oriented opportunistic recognition of high-level composed activities using dynamically configured hidden markov models," in *The 3rd International Conference on Ambient Systems, Networks and Technologies (ANT2012)*, August 2012, pp. 308–315.
- [18] M. Condorcet, *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Paris: Imprimerie Royale, 1785.

# Smart Bin for Incompatible Waste Items

Arnab Sinha

INRIA, Rennes-Bretagne Atlantique,  
Campus Universitaire de Beaulieu  
35042 Rennes Cedex, France  
arnab.sinha@inria.fr

Paul Couderc

INRIA, Rennes-Bretagne Atlantique,  
Campus Universitaire de Beaulieu  
35042 Rennes Cedex, France  
paul.couderc@inria.fr

**Abstract**—In this paper, we present a generalized ontology based system to identify the necessary properties of products and objects and then make inferences of possible incompatibilities between them. This is designed in the context of waste management, where considerable volume of waste is present collectively and there may be chances of hazard or reduces the value of the recyclable waste. These inferences for the group of objects are performed based on the knowledge available locally without looking up from any external sources. Since, the global identification of objects is avoided, preservation of privacy is ensured, which is a concern in the field of pervasive computing. Our model can have applications in various domains. We have demonstrated it's application in the domain of waste management and discussed about other possible domains.

**Keywords**—Radio-frequency identification (RFID); incompatible; ontology; OWL; waste

## I. INTRODUCTION

Today, efficient management and handling of waste are of great importance. The European Union (EU) reports handling around 3 billion tonnes of waste (and still counting) each year among which, around 90 million tonnes are hazardous [1]. So managing the waste i.e. treating and disposing them efficiently causing least possible harm to the environment is becoming difficult. Waste prevention, recycling/reuse and disposal are the three principles laid down by the European Union's approach to waste management. For example, products like packaging waste, batteries, electrical and electronic wastes require special attention for recycling and reuse. Some of these items can be hazardous to be put into a waste bin containing other items.

Sorting is performed for efficient recycling and reuse of the waste materials. Different waste items contain recyclable and reusable materials having economic value. Hence, sorting them at the earliest has advantages. Moreover, it also ensures the quality of the collected waste with no contamination. It also avoids potential hazards. Transporting them to the sorting facilities could be another overhead to cost if not performed early. Lastly they might end up as landfills, if left undetected.

Despite that the early sorting could be beneficial, we need to reason out for the sorting process. Also, care should taken about any possibilities of hazards during this process. The model described in this paper can be used to perform such reasoning.

The rest of the paper is organized as follows: Section 2 presents the background. Section 3 describes the principle for inferring hazardous incompatibilities and how the system is

designed to use the principle in Section 4. A demonstration of our Smart Bin and other possible applications are presented in Section 5. Finally, we discuss the complexity of our system before concluding in Section 6 and 7, respectively.

## II. BACKGROUND

Lately, there has been considerable efforts to increase the cost effectiveness and efficiency of waste management system with the advancement of technology. As described above, the sorting process can be handled efficiently and smartly by using information technologies.

Radio-frequency identification (RFID) is currently an upcoming and rapidly growing mobile technology for the purpose of uniquely identifying and tracking an object attached with a RFID tag. Some RFID tags can be read from several meters away and could be beyond the line of sight. They can also be bulk read and costs from a few cents to a few dollars [3], [4]. These tags are now widely used in various fields like asset tracking, manufacturing, supply chain management, retailing, payment systems, security and access control etc [5].

RFID's have also been considered to be used for better waste management. In RFID and sensor based real-time Automatic Waste Identity, Weight and Stolen Bins Identification System (WIWSBIS), the authors have proposed the use of RFID tags in an environmental context [3]. They have used RFID tags to identify the waste bins, uniquely and remotely. RFID supported waste management and load cell sensor technology are used to automate the billing for customers on pay-per-use basis. While this is a paper where all the information processing has been done at the bin level, we have not come across any work that makes inferences at the item level i.e. based on the contents.

The trend of pervasive computing is transforming more and more everyday used objects smarter with embedded technology and connectivity. *Cooperative Artefacts* is a smart container system where the authors have proposed that the chemical containers are able to assess their situation in the world, without requiring any supporting infrastructure in the environment [10], [11]. The movable artifacts can make rule-based inference based on its embedded domain knowledge and perceptual intelligence. The knowledge is stored in a distributed way across the artefacts. So, communication with the artefacts in proximity might be required frequently to obtain the necessary information before making inferences.

The goal of pervasive computing, which combines current network technologies with wireless computing, voice recognition, Internet capability and artificial intelligence, is to create an environment where the connectivity of devices is embedded in such a way that the connectivity is unobtrusive and always available.

Internet of things (IoT) is a concept where smart devices interact and communicate with other devices, objects, environment and infrastructures. The authors proposes an application for pharmaceutical system for detection of interaction and improper administration of drugs to patients [17]. The drugs are NFC tagged or bar coded for identification and matching with the remote information system. to detect the suitability with patient’s health condition. It works within the IoT paradigm.

In our work, we have proposed an ontology based knowledgebase, which is used to describe the properties and make inferences about the incompatibilities among objects. This knowledgebase is local and powerful enough enabling autonomous decisions. Our objects are passive in nature due to the type of RFIDs used. By contrast, [10], [11] proposes active objects by use of sensors.

### III. HAZARD DETECTION PRINCIPLE

Pervasive computing is the growing trend of progressing beyond the idea of personal computing. Objects used everyday have embedded technologies attached, to perform smarter activities collectively. We have proposed to self describe waste items with their properties using RFID tags. Based on these properties, incompatibilities could be detected among a collection of items present locally. In this section, we discuss its underlying principle. For the purpose we begin with organizing the waste domain in a specific manner for making such inferences.

#### A. Describing waste items

The waste domain can be categorized based on their various hazardous properties. There are standards that specifies the properties of waste materials and categorizes them [2]. Although, discussion on such standards is outside the scope of this paper, however we utilize its idea for categorization and use few examples of hazards related to some of these categories.

Some examples of hazardous properties for this domain are spark, explosion, toxic fumes etc and can categorize based on them. As discussed in the previous section, we are interested to infer incompatibilities. So, it is essential to pick the properties only that are relevant for interactions with other items.

Figure 1 is the pictorial representation of the three conditions under which these properties can act in hazardous ways. They are summarized as below:

- under effect of: the condition(s) which holds the properties that can affect the category
- can cause: this condition enlists the properties that can be caused by the category
- in presence of: this holds the external conditions under which the **can cause** properties occur

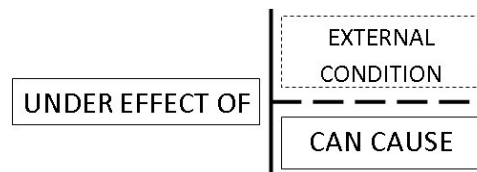


Fig. 1. Conditions to describe a category

In the subsequent sections, we will use the same pictorial representation to describe the waste categories or items in our examples.

Let us take some scenarios of interactions between categories. First, let’s take an example of simple incompatibility between a pair of them. Suppose a category *A* can cause an incidence (for instance say hazardous property *X*) that affects a second category *B*. Hence, an incompatibility exists between the categories *A* and *B*. Our second example is a slightly more complex and realistic than the previous example. If the category *A* causes the incidence (i.e. *X*) only in presence of an external condition (let’s name as *C*), makes it an important augmentation to the scenario. Hence, the categories does not pose to be incompatible if the condition *C* is unfavorable. Both of these scenarios consider the incompatibility between different categories where the hazardous property affects each other. However, there are properties like explosion for example, which have hazardous effect by itself. The situation can be represented as a category that causes a hazardous property that affects itself that may depend on the external condition.

#### B. Inferring incompatibilities

As described above, we can self describe waste items accordingly. When a collection of these items is present locally we can infer incompatibilities based on the discussed scenarios. Sometimes objects are located remotely and communicate within themselves and other knowledgebase using network infrastructure like the Internet to make decisions. Such an idea is called Internet of things (IoT) in the field of pervasive computing. Our approach in this paper proposes making the required information that describes waste domain available locally for inferences. Such collective inferences could be made without using network for communication. We prefer to use the name for such a situation as Intranet of Things (InoT) as it does not involve any devices located remotely and differentiate to avoid confusion.

In III-A, we discussed the interaction scenarios between pairs of categories based on hazardous properties. Multiple such categories can constitute an InoT. The graph in Figure 2 represents an example of InoT formed. The shaded nodes represent some categories. They are connected by an edge if they interact. The dotted edges represent interactions which are unfavorable due to external conditions. One of the external conditions were high temperature at the instance this snapshot was drawn. Hence, the dotted edge encircled in the figure representing an interaction under low temperature becomes unfavorable. The firm edges represents favorable interactions,

which could be either the first or second scenario described in III-A. The shaded node with a self-loop represents the last scenario of III-A, is favorable in this case as the external condition is satisfied.

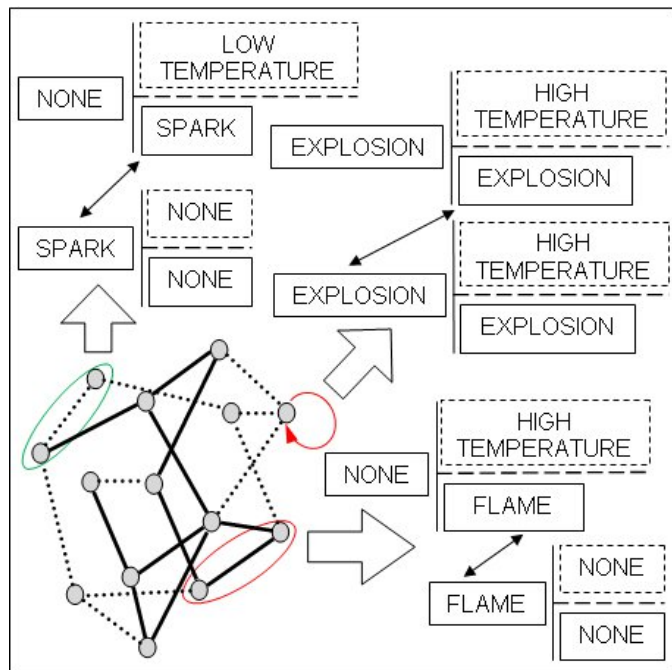


Fig. 2. InoT formed

Finally, if a waste item belongs to one or more categories, it would possess all their conditions. Hence, they could be used for collective inferences also.

#### IV. SYSTEM DESIGN

In this section, we describe designing the system for making inferences locally. It essentially means that all the information required are available without referring to remote database or knowledgebase. An alternative could be to distribute the information partially among the waste items and a local knowledgebase. The waste items are identified by the system before inferring on incompatibilities. We have chosen a commonly used architecture for our system, as shown in Figure 3 below.

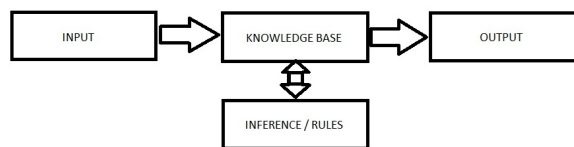


Fig. 3. Commonly used Architecture for Systems

We describe the components briefly.

- **Input:** It is that point in the system where the waste items are identified and added.
- **Knowledge Base (KB):** This contains all the required information to identify the items along with their properties.

It also updates its knowledge regarding the presence of items that are being added to the system incrementally.

- **Inference/Rules:** This component of the model uses the KB to reason out about the possible incompatibilities and hazards. The inferences are added back to the KB.
- **Output:** It sends out notifications to communicate about alerts and warnings to the users of the system.

Next we elaborate on how the system works based on the architecture and uses the principle discussed earlier in Section III.

#### A. Input

New waste items are added to the system. They are affixed with RFID tags only for the purpose of identification by the system, which contains a RFID reader for scanning. The tags do not contain any such data that has privacy concerns. Mostly they contain the category information.

#### B. Knowledge Base (KB)

Machines can be made to perform reasoning effectively provided it has the necessary knowledge, which is machine readable. In cases of large domain knowledge with lots of factors influencing the reasoning, using machines should have extra benefits. Using ontologies are a very good way to serve the purpose [7]. An ontology consists of common set of vocabulary as shared information of a domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them [8]. Lately, the development of ontologies has begun to find many uses outside the Artificial-Intelligence laboratories. They are being commonly used on the World-Wide Web and finds applications for sharing information widely in the field of medicine.

The Web Ontology Language (OWL) is a World Wide Web Consortium (W3C) Recommendation for representing ontologies on the Semantic Web [6]. Presently, there are a lot of ontology editors for OWL. Among them Protégé is a Java based Open Source ontology editor. We used Protégé since we found it to be an efficient and user-friendly tool to prototype our ontology rapidly. During the ontology development phase we visualized the graphical representation of our OWL ontology on the editor. The comprehensive Java API provided by Protégé [12] was also an added advantage while developing our stand-alone application in the later phase.

We have used an ontology based approach for the KB for the reasons stated above. The properties causing incompatibilities must be described in the ontology. Apart from these, other information like conditions in which the categories are incompatible, possible hazards of incompatibility etc are also stored in the ontology.

Due to the advantage for describing a domain easily, we have used ontology based approach for describing the waste domain. The ontology contains description of various categories with the conditions for hazardous properties. This constitutes as the initial knowledgebase of the system, which updates itself as new items are added.

For detailed demonstration, we have used a sample OWL ontology using few hazardous properties, conditions including an external condition to demonstrate the inference of incompatibilities between objects.

We start with building our KB using *Object properties* of OWL ontology that would represent the conditions. As mentioned earlier, a category is described to have various properties under three different conditions. Mapping and comparing Figures 1 and 4 would make the idea very clear.

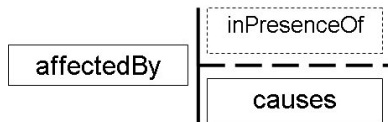


Fig. 4. Object properties as mapped in OWL

Next we define two *Data properties* in the ontology namely *hasStatus* and *hasTemperature*. While the first one can store values of type boolean and acts as a flag, the other is used to hold integer data as in Figure 5. They are used to express the external conditions as explained subsequently.

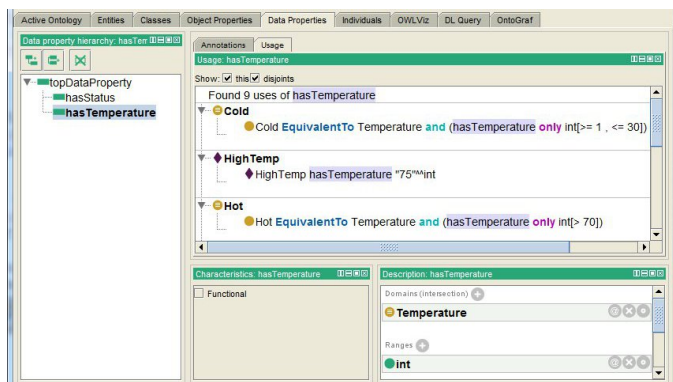


Fig. 5. Data property *hasStatus*

The left side of Figure 6 shows the representation of a domain knowledge using hierarchy of classes in ontology. The class *Waste* is the topmost level in our ontology structure and hence, all its subclasses would represent information of this domain. Following are the description of its subclasses and the information it holds:

- *Properties* list the various instances representing the hazardous properties of the domain. It also includes some external ones like temperature, pressure etc.
- *Categories* represent the classifications of the waste domain. Each of these categories are represented as classes in ontology along with the description of conditions. The RFID tagged waste items added contains reference to these categories. They are added to the system as individuals of the referred subclasses.
- *Hazardous* contain all the incompatible or hazardous items. It's subclass *selfHazardous* holds items that could pose hazardous by itself. They would be subsumed by

the upper class. The user can glance through all the incompatible objects added to the ontology.

The right side of Figure 6 displays subclass *Properties* containing the list of seven properties that are possible with this domain. Examples of OWL individuals such as *Explosion*, *Flame* etc. are some possible properties. *None* represents a special kind of property, which indicates no conditions at all.

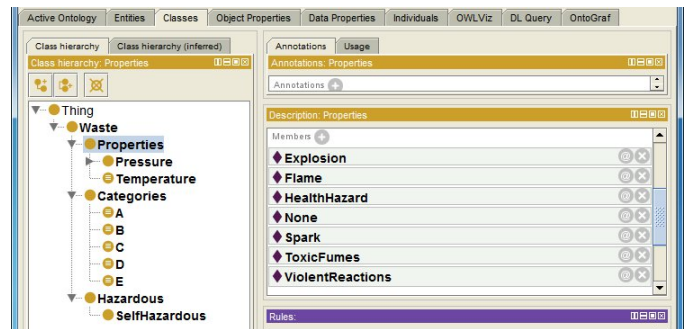


Fig. 6. Sample Ontology with Classes and Conditions

The Figure 7, we have listed examples of five different types of categories named *A, B, C, D, E* subclassed under *Categories* as examples. It should be noted that each of these five classes represent the types of categories possible in the domain described with the conditions of hazardous properties. They are described in form of relationships with individuals of the class *conditions* using the *OWL object properties*.

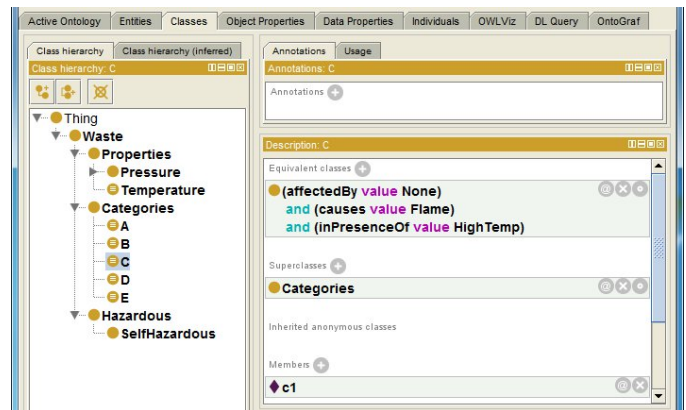


Fig. 7. Describing Categories

In Figure 8, shows an example to represent temperature as an external condition for incompatibility of objects. In this example, we have three individuals of the class *Temperature* as *LowTemp*, *ModerateTemp* and *HighTemp*. They are linked to the two data properties, *hasStatus* and *hasTemperature*. The *hasTemperature* property basically defines the temperature ranges it represents. The *hasStatus* property can be set to *true* for any one of the instances which would indicate that external temperature around the system. Hence, from this way of representation we can indicate discretized levels of some external conditions and the one prevailing around the system.

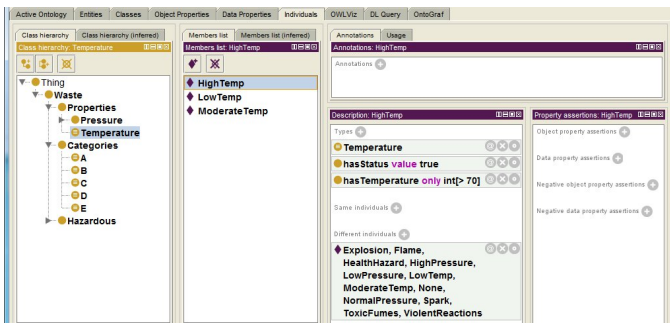


Fig. 8. External Conditions

The class *Hazardous* consists of a subclass *SelfHazardous*. *SelfHazardous* contains all the items that are self hazardous which would be subsumed by the upper class as well. Additionally the class *Hazardous* holds all pairs of incompatible items that are inferred hazardous. In fig 9, we see there are currently one pair of item *c1*, *d1* that exhibit possible incompatibility under an external condition of *HighTemp*.

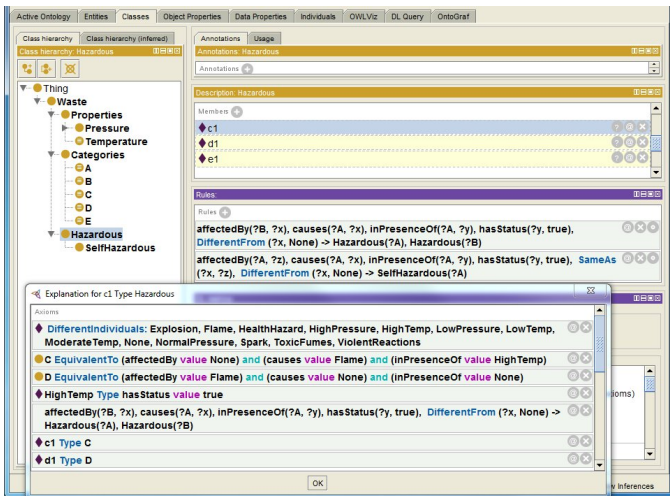


Fig. 9. Dashboard

C. Reasoning/Rules

Given the ontology, which acts as a KB in our architecture, we are all set with having all the necessary information at hand to reason out something useful i.e. the objective of inferring incompatibility or hazards. In the recent years, rule languages have been added on as a layer combined with ontology in order to enhance the reasoning capabilities. Semantic web Rule Language (SWRL) is used to write rules expressed in terms of OWL concepts and for reasoning about OWL individuals. It provides a deductive reasoning specification that can be used for inferring new knowledge from the Knowledge base.

We have used two SWRL rules to make selection of the proper objects and classify them as members of specific class. The first rule is used to classify all the item pairs that may have incompatibility considering if the the external conditions are favorable. In that case they asserted as members of the class

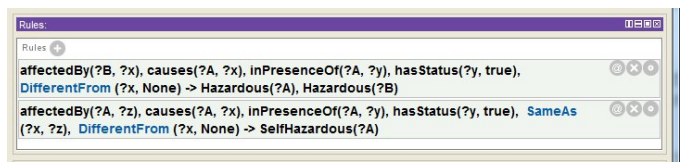


Fig. 10. SWRL Rules

*Hazardous*. The first rule in fig 10 performs this classification. The second rule verifies if an item is self hazardous with favorable external condition. If it's so, the item is asserted as member of class *SelfHazardous*.

V. APPLICATIONS

We have proposed the system using ontology as it's local knowledgebase to infer incompatibilities on the principle of InoT. We think that it can be used to infer incompatibilities among objects in various domains. "Bin That Thinks" is a project, that aims to propose an intelligent waste management solution based on item level identification. The goals are to improve recycling efficiency, reducing waste processing cost and avoiding hazardous situations [16]. Though we have not assessed for the financial benefits figuratively for using our system, our approach hints at the benefits qualitatively. Sorting wastes at the earliest retains the purity of the recyclables. This reduces the cost of sorting at a later stage in processing plants by waste management companies like Veolia, which is usually passed on to the consumers as penalties on the cities.



Fig. 11. Smart Bin

We have developed an application for the domain of waste management using the system described in this paper. It can be used to make inferences for incompatibilities and hazards among the waste items present collectively at a place. They may be situated inside a bin or a waste collecting vehicle or at the processing plant. For very complex domains like waste management, they are sometimes verified at every step in the processing chain. Alternatively, when the processing is performed at a single point, we consider the acceptance of

error up to some limit. Fig 11 shows the smart bin developed that can identify the RFID tagged wastes and make inferences from its contents. In fig 12 below shows a screenshot of our application. It shows the instance when an incompatibility is detected with two items present locally in the bin and the last item that was scanned. It also displays the inferred reasoning.

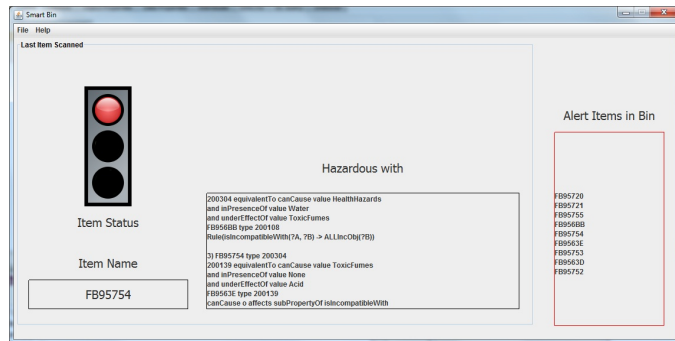


Fig. 12. Hazard Detection Application for Waste

Another domain of application for our model can be in the field of medicine. Storing medicines together can sometimes be potentially dangerous. It might also lead to confusion and take wrong medications. The elderly people and children are more vulnerable to such mistakes. Also some medicines might react with each other ('interact') if taken together and might cause serious problems.

VI. COMPLEXITY

The OWL from the W3C has capabilities to describe concept from very simple to quite complex ones. It provides with a variety of features to express some domain of interest. OWL ontology has three different types of sub-languages namely OWL-Lite, OWL-DL and OWL-Full. These sub-languages differ in the amount of features incorporated in it and hence, have varying degrees of expressiveness. W3C provides a description of the features to be used for OWL-Lite compared to OWL-DL or OWL-Full [6]. The profile for our model gets disqualified from being OWL-Lite as we have used OWL individuals to occur in descriptions or class axioms i.e. by using the value constraint owl:hasValue [6], [13], [15]. The OWL-Lite has computational complexity of polynomial order whereas the rest grows exponentially [14]. So, it essentially means that the computational complexity of our model can be calculated to grow exponentially with increase in the ontology size. This is important in the context of pervasive computing as the setup would be functioning on embedded computers, which have limited memory and computational capabilities.

VII. CONCLUSION

In this paper, we demonstrated a system to infer the incompatibilities between collective waste items. As discussed earlier, the model can have applications in various domains. Presently, we have designed to make inferences particularly for the domain of waste management. In the context of waste

management, the originality of our approach consists in representation and processing of knowledge, and make inferences at the item level, rather than container or bin level. And more importantly, this can be done locally without referring to external knowledge base. Privacy is also maintained in spite of using RFID tags containing category information of the item itself. This is a concern for pervasive computing applications. Our future direction would be to fine-tune the ontology further, so that the complexity remains in the order of polynomial time. Also, we are interested to keep greater amount of distributed information to make our inferences better, precise and more scalable. An approach would be putting more semantic data into the RFID tags to describe the item as we have proposed in [18]. It would make the system more distributed, thus reducing the information in the knowledgebase and also the dependency on it.

REFERENCES

- [1] "The EUs approach to waste management", available: <http://ec.europa.eu/environment/waste/index.htm> [accessed 16 January 2013, 20h19].
- [2] "European waste catalogue and hazardous waste list", available: [www.environ.ie/en/Publications/Environment/Waste/WEEE/](http://www.environ.ie/en/Publications/Environment/Waste/WEEE/) [accessed 14 March 2013, 10h28].
- [3] B. Chowdhury and M.U. Chowdhury, "RFID-based real-time smart waste management system", Telecommunication Networks and Applications Conference, 2007. ATNAC 2007. Australasian, 2-5 Dec. 2007, pp.175-180.
- [4] S. A. Weis, "RFID (Radio Frequency Identification): Principles and Applications", MIT CSAIL, 2007.
- [5] Association for Automatic Identification and Mobility, "What is Radio-frequency identification (RFID)?", available: [http://www.aimglobal.org/?page=rfid\\_faq](http://www.aimglobal.org/?page=rfid_faq) [accessed 16 January 2013, 22h16].
- [6] World Wide Web Consortium (W3C), "OWL Web Ontology Language Overview", available: <http://www.w3.org/TR/owl-features/> [accessed 16 January 2013, 22h25].
- [7] C. Mathieu, E.S. Eltaher, P. Sarathy, and S. Gilles, "OWL Ontology for Solar UV Exposure and Human Health", Advances in Semantic Computing, Eds. Joshi, Boley & Akerkar, Vol. 2, 2010, pp 32-51.
- [8] T. R. Gruber, "A Translation Approach to Protoble Ontology Specifications", Knowledge Acquisition, 5(2), 1993, pp. 199-220.
- [9] N. F. Noy and D. L. Mcguinness, "Ontology Development 101: A Guide to Creating Your First Ontology", Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory, 2001.
- [10] M. Strohhach, G. Kortuem, and H. Gellersen, "Cooperative artefacts: a framework for embedding knowledge in real world objects", Smart Object Systems Workshop, UbiComp 2005.
- [11] M. Strohhach et al., "Cooperative Artefacts: Assessing Real World Situations with Embedded Technology", Proc. Int'l Conf. Ubiquitous Computing (Ubicomp 04), Springer, 2004, pp. 250267.
- [12] H. Knublauch et al., "The Protégé OWL Experience", Proc. OWL: Experiences and Directions Workshop 2005.
- [13] World Wide Web Consortium (W3C), "OWL Web Ontology Language Reference", available: <http://www.w3.org/TR/owl-ref/> [accessed 16 January 2013, 22h10].
- [14] C. Maria Keet and M. Rodriguez, "Toward using bio-ontologies in the Semantic Web: trade-offs between ontology languages", Proc. of the AAAI Workshop on Semantic e-Science (SeS 2007), 2007, pp 65-68.
- [15] F. Baader, "The description logic handbook: theory, implementation, and applications", Cambridge Univ Pr, 2003, pp. 479.
- [16] Bin That Thinks, available: <http://binthatthink.inria.fr/> [accessed 16 January 2013, 20h35].
- [17] A.J. Jara, A.F. Alcolea, M.A. Zamora, A.F.G. Skarmeta, and M. Alsaedy, "Drugs interaction checker based on IoT," Internet of Things (IOT), Nov. 29 2010-Dec. 1 2010, pp 1-8, doi: 10.1109/IOT.2010.5678458.
- [18] A. Sinha and P. Couderc, "Using OWL Ontologies for Selective Waste Sorting and Recycling", 9th OWL: Experiences and Directions Workshop, OWLED, 2012.

## Middleware Supporting Net-Ready Applications for Enhancing Situational Awareness on Rotorcraft

Gustavo L. B. Baptista, Markus Endler  
Department of Informatics  
PUC-Rio  
Rio de Janeiro, Brazil  
gbaptista, endler @inf.puc-rio.br

José Viterbo Filho  
Computing Institute  
UFF  
Niteroi, Brazil  
viterbo@ic.uff.br

Thomas A. DuBois  
Technical Fellow – Software, Avionics and Systems  
The Boeing Company  
Ridley Park, PA  
thomas.a.dubois@boeing.com

Randall L. Johnson  
Associate Technical Fellow – Avionics Software  
Bell Helicopter Textron  
Fort Worth, TX  
RJohnson@bellhelicopter.textron.com

**Abstract**—This work presents middleware to support network ready applications that enhance Situational Awareness (SA) on rotorcraft operations. SA of pilots, command and control teams and other participants collaborating in missions, is critical for rotorcraft operational performance, mission success, safety and survivability. Applications and systems that aim to enhance SA impose requirements of real-time communication and processing of data streams with high throughput and low latency. The presented middleware provides capabilities such as real-time data-centric publish-subscribe communication with Quality of Service (QoS) contracts, Semantic Interoperability, and Distributed Complex Event Processing for real-time detection of situations. An illustrative Situation of Interest (SoI) is presented, based on a realistic scenario of helicopter rescue missions for offshore drilling. Simulation results are presented that compare different event processing distribution models and abstraction levels, regarding their impact on performance and scalability. \*

**Keywords**—middleware; data distribution service; complex event processing; data stream processing; situational awareness.

### I. INTRODUCTION

Mission critical scenarios, typical of rotorcraft operations, demand effective communication, coordination,

---

\* This project was funded by the Vertical Lift Consortium, formerly the Center for Rotorcraft Innovation and the National Rotorcraft Technology Center (NRTC), U.S. Army Aviation and Missile Research, Development and Engineering Center (AMRDEC) under Technology Investment Agreement W911W6-06-2-0002, entitled National Rotorcraft Technology Center Research Program. The authors would like to acknowledge that this research and development was accomplished with the support and guidance of the NRTC and VLC. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the AMRDEC or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

decision-making and response under temporal and resource constraints. The burden of the workload on pilots associated with flying rotorcraft, conducting difficult missions with operational risks and environmental hazards [1] has to be lightened by tools that provide useful and instantaneous insight over many aspects and dimensions.

### A. Situational Awareness

Situational Awareness (SA), a term coined in the aviation and military domains, according to [2], is “the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future”. In order to enhance such awareness, systems shall capture real-world events and entities (i.e., by using sensors), communicate them to all interested parties, perceive and process them to detect situations that are relevant to the mission at hand. Since the elements to be perceived are both local to rotorcraft (e.g., local health and monitoring systems, geographical location and many other types of context data on a helicopter) but also spread out over other mission participants and the environment, a large set of sensors and communication technologies have to be combined.

### B. Net-Ready Applications

Emerging communication technologies, such as IP-based radio Mobile Adhoc Networks (MANETS), which are becoming mature enough for use on rotorcraft, publish/subscribe middleware and common operating picture displays are being used as base for the development of network-ready (a.k.a. net-ready) applications that result in better pilot SA for improved safety and survivability [1]. For example, these applications provide the pilots with more awareness of the current location of other aircraft, vehicles, objectives, weather, uncharted obstacles that may impact the mission, and aids for difficult landing situations.



### C. Coordination Models

SA is critical for safety and survivability of rotorcraft, but it also plays a larger role in establishing means for more effective coordination and collaboration between many participants involved in a flight or mission [3]. In centralized command and control, a control station typically receives information about the monitored assets in real-time, allowing rapid decisions and actions to be taken by control teams, such as modifying objectives, new targets, re-routing of task-force team members, and other major operational or tactical changes in a mission. Another way to allow timely collaboration among the team members is to support decentralized, network centric mission communication and coordination. In this approach, each individual agent of the team performs its specific mission, but continuously shares data as needed with others so as to provide a more complete and accurate representation of the environment and better define its current role in a task force group. This type of coordination appears suitable for emergency search-and-rescue missions (e.g., natural disasters) or tactical coordination (e.g., police or military patrol). In fact, such centralized and decentralized types of coordination can both be supported by command-and-control and decentralized collaboration systems at the same time, depending on the types of scenarios and missions.

### D. Situations of Interest (SoIs)

In order to address SA enhancements, it is necessary to have the scope of specific Situations of Interest (SoIs) that are important for the accomplishment of missions. We consider as a Situation of Interest (SoI) a description of a spatio-temporal condition with regard to the context of monitored elements (e.g., their geographical position) and/or the environment state (e.g., the weather condition). The detection of a SoI usually demands a notification to interested parties (e.g., users or systems), either because it represents a desirable or undesirable/dangerous overall situation. A SoI can have a global or local focus, which can be respectively regarded as global SoI or local SoI. An example of global SoI is providing control station operators (or pilots on rotorcraft) overall situations related to a group of nodes (e.g., too many/too few helicopters in a region). An example of local SoI is showing a pilot combined local information from remaining fuel, weather data, obstacles and mission objectives.

### E. Visualization

Visualization techniques play a central role in enhancing SA [4]. In the context of military and rescue missions, it generally involves the representation of geographic areas in maps or 3D environments, with the elements that are important to be monitored and considered for decisions in missions, or aiding individuals to perform specific tasks, for example, providing visual signals to a pilot during landing approaches. Appropriate visualization requires showing users the right information, at the right time, building a common picture of the operational area. Since the complexity and potential large number of elements and data sources (e.g., sensors and monitored nodes) can cause clutter

to visualization, showing aggregations that combine large amounts of raw data is necessary.

### F. Goals

In order to implement systems that support coordination and visualization enhancing SA, support is necessary from underlying middleware with mechanisms for real-time communication and processing of data streams with high throughput and low latency. In this work, we present a middleware architecture with such features, giving focus to its capabilities that allow the real time monitoring of nodes (e.g., rotorcraft) and the evaluation of general relations among mobile and stationary nodes, or virtual entities (e.g., arbitrary geographical points), in a scalable manner. The work in this paper has been developed in the context of a project called “Net-Ready Applications to Improve Rotorcraft Safety and Survivability”, sponsored by The National Rotorcraft Technology Center / Vertical Lift Consortium (NRTC/VLC). The goals of this project are to make rotorcraft more safe and survivable through the application of emerging net-ready technologies.

After the description of base technologies in Section II, the middleware architecture is presented in Section III. A realistic scenario of helicopter emergency response missions in the offshore drilling domain is presented in Section IV, and Section V describes the application of this architecture to the context of the presented scenario. Performance results are then presented in Section VI that compares simulations with different event processing distribution models and granularities for data abstraction levels, regarding their impact on scalability and performance.

## II. BASE TECHNOLOGIES

This section presents the technologies for event-based communication and processing, which have been used as base for the middleware presented in this work.

### A. OMG Data Distribution Service (DDS<sup>TM</sup>)

The Data Distribution Service for Real-Time Systems (DDS<sup>TM</sup>) [5] standard determines a peer-to-peer (i.e., fully distributed and without broker nodes) data-centric publish/subscribe communication model. It provides high performance real-time communication, scalability and availability, and supports the specification of Quality of Service (QoS) contracts between data producers and consumers. It allows interoperability across different DDS implementations, programming languages and platforms, as well as automatic discovery of DDS publishers/subscribers. DDS is based on a Data Centric Publish/Subscribe model, where DDS Topics are logical entities defined to compose a distributed relational data model, also known as Global Shared Data Space. The Topics are the first class entities of information, which applications can publish or subscribe to, and can be regarded as distributed relational database tables. The DDS Domain, which contains all shared data, is fully distributed over the participating network nodes, without any intermediate broker or centralized management entity.

Several commercial and open-source implementations of DDS are available, such as [6], [7].

*B. Distributed Complex Event Processing*

Complex Event Processing (CEP) [8] extends the capabilities of the content-based publish subscribe model [9], with the capacity to specify relationships not only over event properties, but also relationships between different events, causality, temporality, sequencing, aggregation and composition. A complex event is thus a higher-level abstraction representing a situation derived from the occurrence of more elementary events. Causal maps can be associated to these events, allowing a complete tracking of the event causes (i.e., the sequence of events that caused an event to happen). The temporal relationships between events allow the processing of event sequences within specified time windows. It is also possible to evaluate aggregation functions over event properties observed in sets of events, such as the average, maximum or minimum property values of the observed event set. All these features allow the definition of powerful event processing rules that express application-relevant patterns of events and their relationships [10]. Many commercial and open-source implementations of CEP are available, such as [11]–[15].

*1) Event Processing Agents and Event Processing Networks*

An Event Processing Network (EPN) is a conceptual representation of an event processing system in a platform independent way. An EPN is composed of Event Processing Agents (EPAs), which receive event streams as input and process these events with different operations (e.g., filtering, aggregation, transformation, pattern detection, etc) producing events as output, as the result of this processing (The concept of Event Processing Agent is not related to, and shall not be confused with, the concept of “agent” in Agent-Based Models or Multi-Agent Systems). In an EPN, the EPAs are conceptually connected to each other (i.e., output events from one EPA are received and further processed by other EPAs), without regard to the particular details and type of the underlying communication mechanism (e.g., push- or pull-based) used to transfer events between each other. The EPAs organized in an EPN implement the whole processing logic of situation detection through event processing [8], [16].

*2) EPNs Distribution Models*

Event processing systems (e.g., CEP systems) implement the concepts of EPNs in slightly different ways, but one important aspect that has the largest impact on scalability is the deployment model of the EPN (centralized vs. distributed). The distribution of the event processing architecture is a key aspect to allow scalability on the number of data producers and data consumers, and also scalability in the number of SoIs to be detected from large amounts of events flowing through the system [17]. Different academic and commercial distributed event-processing systems are available. The majority of them use a distributed clustered deployment model, e.g., [11], [12], [15], with few

implementing a distributed networked deployment model [13], [18].

*C. SoIs Expressed as CEP Rules*

The capabilities of processing event streams and the recognition of event patterns on real time provided by the CEP model make it very well suitable for the detection of SoIs, to enhance SA. Rules can be deployed on EPAs to compose EPNs, which trigger complex events representing the detected SoIs. These complex events can be captured by applications, systems or used by other rules to detect many abstraction levels of SoIs.

III. MIDDLEWARE ARCHITECTURE

In the context of this NTRC/VLC project, a set of capabilities for applications was prioritized for implementation, all with a network-centric approach for real-time data sharing among rotorcraft nodes [1], [19]. Middleware APIs, encapsulating the use of DDS for data-centric publish/subscribe communication, were developed supporting capabilities, such as Networked Weather, Uncharted Obstacles, Own Ship Position Reporting, and Aids for Landing Operations. The API methods allow applications to interact with the DDS Global Shared Data Space synchronously or asynchronously, without requiring the application programmer to deal with DDS-specific entities or to have deep knowledge about the DDS specification. Figure 1 shows an overview of the Middleware for Net-Ready Applications, with the communication APIs for the different net-ready applications, and modules for managing QoS, Semantic Interoperability and Distributed Complex Event Processing, explained in sections A, B and C respectively. We implemented the middleware for OpenSplice DDS [6] and RTI Connex DDS [7] commercial DDS products, in C++ and Java. The DCEP module is currently implemented in Java with instances of the Esper CEP open-source engine.

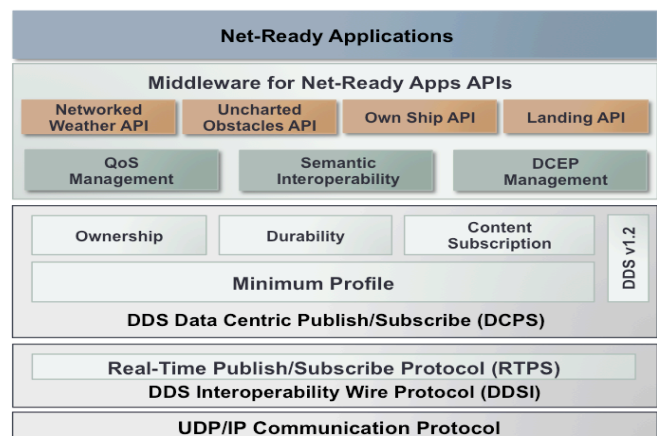


Figure 1. Middleware for Net-Ready Applications Overview, built on top of a DDS implementation.

*A. Quality of Service Management*

The DDS specification provides a rich set of Quality of Service (QoS) parameters, features and enforcement

mechanisms (e.g., communication reliability, latency, transport priority, data persistence, etc.). However, the configuration and association of these QoS parameters with the different DDS entities (i.e., Domain participant, Publisher, Subscriber, DataReader/Writer and Topics) is usually restricted to be done at application development time, and is work-intensive. Therefore, a QoS Management API was developed to provide direct means of defining DDS QoS parameters dynamically through configuration files, which also serve as QoS templates that can be used by the other components of our middleware.

Using the QoS Management API, important QoS settings can be set, depending on the scenario of usage, for example: Defining prioritized event flows for meeting real-time requirements, in accordance to the level of priority required by SoIs. Setting the level reliability for data delivery to ensure delivery or the discarding of certain types of events in the case of network failures. Setting the persistence level of different Topics to allow late joiners to receive events already published into the system, or to make data volatile.

### B. Semantic Interoperability Support

When systems operated by different groups, belonging to different organizations, need to interact and exchange information, the use of different data standards and formats may become a hurdle. Although much shared data are conceptually of same type, e.g., position, speed, cargo weight, etc., they may be represented in different units and formats. In the specific case of geographic location, it may be described by different projection, datum and coordinate systems [20]. To tackle this problem, we implemented a semantic interoperability service, where a node in the DDS Domain assumes the mediator role. Whenever a new node joins the system, it informs the mediator which model it adopts to represent the data to be shared. This mediator is capable of querying an Ontology Manager to obtain further semantic information about how to deal with different data models adopted by the peers. The Ontology Manager stores ontologies that represent known semantic models and conversion rules between them, and delivers these rules to the mediator. As to the data conversion process, it may be performed by two different approaches. In a centralized approach, upon receiving the conversion rules from the Ontology Manager, the mediator will be responsible for converting all data among the different models. Of course, this is only feasible if the amount/frequency of data exchange is low. In a translation-on-receiving approach, the mediator will forward the conversion rules to all nodes, and each one will produce data using its original model. When a node then receives data represented in a different model, it will itself perform the translation.

### C. Distributed Complex Event Processing Management

The Distributed Complex Event Processing Management (DCEPM) module, as shown in Figure 1, implements the distributed complex event processing architecture, which allows the creation of EPNs to detect SoIs. The EPAs use DDS to send and receive events from/to monitored nodes, and to exchange events with each other. In the former case,

the middleware APIs for Net-Ready Applications, in addition as being used by applications, are used by the EPAs to subscribe and publish to the DDS Topics that share data from monitored nodes and applications. On the latter case, the EPAs provide and use additional DDS Topics that are specific for them, which can also be used by applications to consume events.

Each EPA contains a DDS Subscriber for subscribing to events from the desired sources. For example, an EPA can subscribe to events from rotorcraft nodes or other EPAs. A CEP engine is instantiated inside each EPA, with deployed rules that process events received from the DDS layer. Since the communication APIs are separated from the CEP Engine, EPAs can use any desired internal CEP engine for processing events (the current implementation uses the open-source Esper CEP engine). An internal DDS Publisher allows the EPA to publish events into the DDS Domain, after processing input events with the internal CEP Engine.

### D. Net-Ready Applications With Visualization

The middleware presented in this work provides many benefits for the development of applications that enhance SA in rotorcraft missions. We developed applications with capabilities for monitoring Own Ship positions, Weather Reports and Obstacles, showing a common operational picture on a map (using instances of FalconView [21] and Google Earth™). In [1] we presented details about the development of those applications, and in Section VII we show simple functional evaluation with a visualization tool to show the detection of a Sol. Several other benefits are provided to applications from the decoupling, interoperability and QoS management provided by this middleware. Network ready visualization applications can be quickly adapted to select and present data of interest particular to the mission or aircraft. Implementation across varied operating systems and visualization systems provides broader accessibility to different aircraft cockpits and avionics suites [1]. This allows interoperability between civilian and military providing similar views of the emergency scenario to all enabled responders. QoS settings and security implementations can segregate unnecessary or sensitive information from individual responders. If a responder is a rescue unit, it could filter on medical needs, landing area obstacles, weather, etc. Fire suppression units could filter on reported fire zones, wind, and elevated obstacles. Security forces could filter on possible incursion treats from outside the evacuation zone, etc. Each individual responder, depending on their current role, could select preconfigured event sets or custom event sets. Tactical command centers observing all operations could quickly respond to changing conditions.

## IV. REPRESENTATIVE SCENARIO AND SITUATION OF INTEREST

In Brazil, the offshore regions of the states of Rio de Janeiro, São Paulo and Espírito Santo are areas of intense oil exploration, and are responsible for up to 80% of the Brazilian oil production. Therefore, many offshore oil drilling and extraction platforms are deployed in this region,

and can be located as far as 100km from the coast. Because of this exploration activity there are ships and helicopters transporting employees, cargo and equipment between land bases and the offshore platforms. However, both the oil extraction activities of the offshore platforms, and the traffic of ships and aircraft impose operational risks. Employees working on oil platforms are exposed to risks of fire or explosions and the inherent risks of helicopter air transportation. Several emergency incidents occurred during the last several years, such as fire and explosions on platforms. Also, helicopter crashes have been reported. The large distances these helicopters have to fly, without any close emergency landing spots, expose them to unexpected and severe weather and also make them vulnerable to mechanical failures. These risk factors and recent incidents show that the capacity of effectively performing emergency response operations in these regions is very important, to respond to incidents involving oil platforms, ships and helicopters.

### A. Situation of Interest

In order to show how the proposed system is able to support applications that enhance SA, we describe a hypothetical Sol to be detected by the system. The example Sol is characterized by the generation of an alert notification when a significant percentage of monitored nodes is out of range from a set of previously defined stationary set of Points of Interest (PoIs). An application of this inference could be, for example, to detect when a set of helicopters, which cover routes between support bases (e.g., land bases, oil extraction platforms, military support bases, etc.), are too far from all these points at the same time, characterizing an exposure to high operational risk. For instance, if rescue helicopters are all away from support bases at the same time, an unexpected emergency situation that requires their reallocation can be difficult to manage. Applying this situation to the aforementioned scenario, we define that the helicopters are the monitored nodes, and the land bases are the PoIs. Considering all the operational risk involved, it is important to have a minimum number of rescue helicopters close to support bases, so they can return and reload with the required resources (e.g., fuel, medicine or cargo). This Sol was chosen because it presents characteristics that allow us to explore its detection with multiple granularities of event abstraction levels (explained in Section III).

## V. DISTRIBUTED COMPLEX EVENT PROCESSING INSTANTIATION

In order to implement the detection of the Sol described in Section IV.A, an EPN with a specific hierarchical topology is used. We define three main kinds of EPAs, as shown in Figure 2. Node EPAs are deployed in the mobile monitored nodes, containing CEP rules (i.e., event correlation and patterns rules) that detect primitive events locally at the node (e.g., from local sensors), and generate events that represent abstractions of situations detected locally on that node.

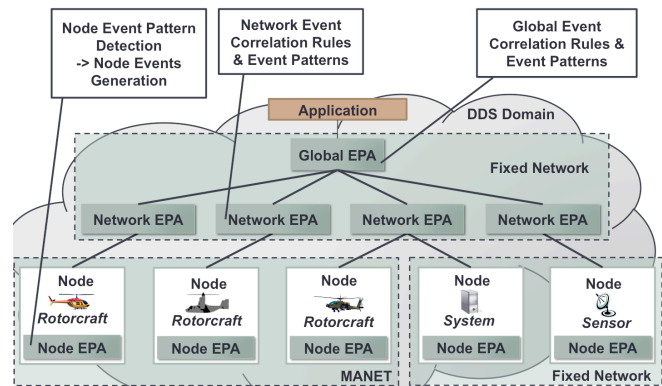


Figure 2. Example of an EPN Hierarchical Topology with Node EPAs, Network EPAs, and a Global EPA.

Network EPAs are deployed in the fixed network and receive events from a set of Node EPAs in their responsibility (the sets of Node EPAs in the responsibility of a Network EPA can be defined following any desired criteria, like geographical regions). The Network EPAs contain CEP rules that detect events related to all the nodes in their responsibility. The Network EPAs process the events received from Node EPAs and generate events with a higher level of abstraction (e.g., summarizations, aggregations of primitive events). Global EPAs are also deployed in the fixed network, and receive events from a set of Network EPAs. The Global EPA contain CEP rules that detect events related to all Network EPAs, and produce events with an even higher level of abstraction/aggregation, that characterize the detected overall Sol. This hierarchical organization of agents is general enough to be used for the detection of many other similar Sol's, since it reflects an organization that promotes scalability. Event Abstraction Levels Different granularities and abstraction levels can be used to define CEP events, and there's subjectivity on this choice. For example, considering the Sol presented in Section A, the Own Ship Position Report event sent by rotorcraft nodes is considered to be at a fine-grained abstraction level. Other events considered as having a coarser-grained abstraction level (i.e., generated by aggregation of events with lower abstraction levels) can be defined. On our example situation, a node has knowledge about existing PoIs, and calculates if it is within a defined distance range from all of them. So instead of sending fine-grained Own Ship Position Report events, it can send a coarse-grained event representing whether the node is within or out of range from PoIs, called "RangeStateFromPoIs", bringing to the monitored nodes the processing effort, and reducing the load on the Network EPAs. Section VI shows performance results of simulations with different abstraction levels of events exchanged between processing agents, and how they impact performance and scalability.

## VI. PERFORMANCE RESULTS

This section describes tests performed with the detection of the Sol presented in Section A, to assess how using multiple distributed EPAs may improve overall scalability in

the rate of events processed by the system, comparing to the use of a single centralized EPA. The tests compare different settings of events abstraction levels and EPN distribution models, to assess how they influence performance and how they scale in the number of monitored nodes and PoIs. Although the SoI presented in Section A does not require processing a large number of PoIs (as the number of offshore platforms and land bases is not large), other different SoIs may require the processing of a large set of virtual entities (e.g., an aircraft monitoring obstacles on real time). Two settings for the abstraction levels of events were used; (A1) Fine-grained Abstraction of Events – Rotorcraft nodes send Own Ship Position Report events (with their current position), and Network EPAs perform all processing to verify which nodes are out of range from PoIs. (A2) Coarse-grained Abstraction of Events - Each rotorcraft node has knowledge of the geographical coordinates of all PoIs and publishes a coarse-grained abstraction event indicating whether or not it is on range from all PoIs at a given moment (“RangeStateFromPoIs”). This local processing removes the processing burden from the Network EPAs, transferring it to the nodes to promote overall system scalability. Two settings for deployment models of EPNs were used: (B1) Centralized Event Processing - All monitored nodes send events to a single Network EPA. (B2) Distributed Event Processing - Monitored nodes send events to distributed Network EPAs. Each Network EPA processes events from nodes in its responsibility and disseminates consolidated reports (i.e., coarse-grained events) with this data. A Global EPA aggregates data from all Network EPAs, counting the overall percentage of nodes out of range from PoIs, and generating an event indicating the global situation. The tests were performed with an infrastructure of 6 machines interconnected in a Local Area Network (LAN) by a Gigabit Ethernet switch. Up to 3 Network EPAs and 1 Global EPA, were deployed in four different machines with Quad-Core Intel i5 processors and 8GB RAM, running Fedora Linux 15, 64 bits. A load generator application, simulating rotorcraft nodes, Node EPAs, and an instance of Google Earth™ application, were respectively deployed in two laptop computers. For each setting (A1, A2, B1 and B2), 1000, 2000 and 3000 rotorcraft nodes, and 100, 500 and 1000 PoIs were simulated. A maximum range distance from PoIs of 10 kilometers was specified, and each rotorcraft node was positioned in a fixed location outside the range of all POIs. Figure 3 shows a chart comparing the throughput of the system for the different settings (A1, A2, B1 and B2). As expected, the coarse-grained abstraction of events removed the processing effort from the Network EPAs, improving the overall system throughput. Since each monitored node sends data to a specific Network EPA, the overall system average throughput is the sum of the average throughput of all Network EPAs.

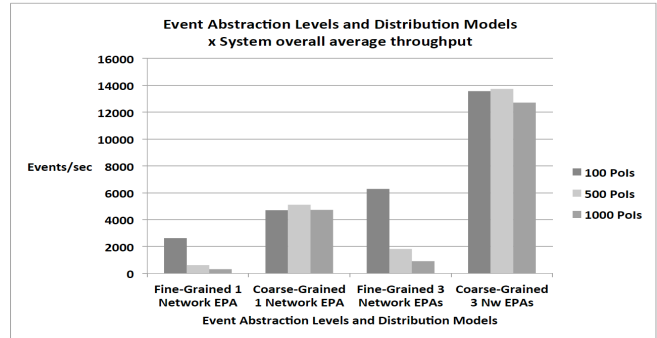


Figure 3. Overall system average throughput with different event abstraction levels and EPN distribution models

This demonstrates the benefits of the distributed deployment of Network EPAs. Regarding network bandwidth usage (not addressed in this test), it is reasonable to assume that the periodicity of events with coarser-grained abstraction levels is in general lower than the frequent sending of events with finer-grained abstraction levels. In applications where the fine-grained events are still necessary for other inferences or system functionalities (e.g., the Own Ship Position is necessary for a control station to show the location of nodes in a map), these events will be propagated through the DDS domain peer-to-peer network only to the interested parties, not affecting the other nodes.

## VII. VISUALIZATION

This work is primarily focused on the middleware aspects to support applications and visualization techniques at a higher level and it is not the goal of this paper to evaluate the level of SA provided to users, or its actual impact on operational performance. In [1], we presented a measurement on the value of integrating the underlying communication infrastructure and the network-ready applications, on rotorcraft for improving operational performance. A test with a visualization tool was used to serve as a functional evaluation of the capabilities provided by the underlying middleware architecture. In the context of the representative scenario presented in Section IV, we illustrate the use of a visualization tool which benefits from the middleware architecture, and how it shows the example SoI. Figure 4 shows a screenshot of a Google Earth™ application, showing the coast of Rio de Janeiro state and rotorcraft nodes (represented in MIL-STD-2525A symbology) flying between land bases and offshore platforms. The nodes in green are inside the specified range from PoIs (i.e., land bases), and the ones in red are out of range from PoIs.

## VIII. RELATED WORK

The use of middleware to support applications that enhance SA (e.g., visualization applications) is addressed by other work, such as [4] and [22]. Feibush et al. [4] present a visualization tool and a middleware architecture, based on CORBA for client-server communication between monitored

nodes and an infrastructure of services. Commercial solutions, like Solipsys Tactical Display Framework (TDF) [22] provide middleware and advanced visualization solutions for many types of mission-critical systems. In general, the integration of pure visualization applications (e.g., FalconView [21]) with any type of middleware has the potential to benefit users with enhanced SA. Many different academic and commercial middleware implementations provide event-based communication and distributed event processing, such as [11]–[15], [17], [18], [22].



Figure 4. Real-time visualization, showing simulated rotorcraft nodes

The benefits of our middleware architecture come from the combination of using real-time data-centric publish-subscribe communication model with advanced QoS features provided by the DDS Specification, Semantic Interoperability, and Distributed CEP, provided by the modules and APIs presented in this work. To the best of our knowledge, none of the distributed event processing middleware solutions combines all these features. The middleware presented in this work can be integrated in any desired visualization tool or framework, and the use of Google Earth™ was due to its simplicity in providing an initial experimentation.

### IX. CONCLUSION

In this work, we presented a middleware architecture to support net-ready applications that can enhance SA, with capabilities such as real-time peer-to-peer data-centric publish/subscribe communication, QoS Management, Semantic Interoperability and Distributed Complex Event Processing. A representative scenario and a SoI instantiation to be detected by the system were presented. Simulations with the detection of the SoI were performed, and a visualization tool was used to illustrate the capabilities of the proposed architecture, in the context of the representative scenario. Performance results show that the distribution of EPAs and the use of coarse-grained event abstractions improve the overall system scalability regarding the throughput in processing events. The use of coarse grained event abstraction levels partially removed the processing load from the infrastructure, taking advantage of the processing power present in nodes, and also potentially reducing network bandwidth usage, which is very important in resource constrained mobile networks.

### REFERENCES

- [1] T.A. DuBois, B. Blanton, F. Reetz III, M. Endler, W. Kinahan, G.L.B. Baptista, and R. L. Johnson, “Open Networking Technologies for the Integration of Net-Ready Applications on Rotorcraft,” Annual conference of the American Helicopter Society. 2012.
- [2] M. R. Endsley, “Toward a theory of situation awareness in dynamic systems,” *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, no. 1, pp. 32–64, 1995.
- [3] D. S. Alberts, J. Garstka, and F. P. Stein, *Network Centric Warfare: Developing and Leveraging Information Superiority*. Washington DC: C3I Command and Control Research Program, 2000.
- [4] E. Feibush, N. Gagvani, and D. Williams, “Visualization for situational awareness,” *Computer Graphics and Applications, IEEE*, vol. 20, no. 5, pp. 38–45, 2000.
- [5] OMG, “Data-Distribution Service for Real-Time Systems (DDS).” 2006.
- [6] PrismTech, “OpenSplice DDS.” [Online]. Available: [www.prismttech.com](http://www.prismttech.com) [retrieved: January, 2013].
- [7] RTI, “RTI Connxt DDS.” [Online]. Available: [www.rti.com](http://www.rti.com) [retrieved: January, 2013].
- [8] D. C. Luckham, *The power of events: an introduction to complex event processing in distributed enterprise systems*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2001.
- [9] P. T. H. Eugster, P. A. Felber, R. Guerraoui, and A. M. Kermarrec, “The Many Faces of Publish/Subscribe,” *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.
- [10] G. Cugola and A. Margara, “Processing flows of information: From data stream to complex event processing,” Technical report, Politecnico di Milano, 2010.
- [11] SYBASE, “SAP Sybase Event Stream Processor CEP.” [Online]. Available: <http://www.sybase.com/products/financialservicesolutions/complex-event-processing> [retrieved: January, 2013].
- [12] Esper, “Esper.” [Online]. Available: <http://esper.codehaus.org/> [retrieved: January, 2013].
- [13] TIBCO, “TIBCO Business Events.” [Online]. Available: <http://www.tibco.com/products/event-processing/complex-event-processing/default.jsp> [retrieved: January, 2013].
- [14] Oracle, “Oracle CEP.” [Online]. Available: <http://www.oracle.com/technetwork/middleware/complex-event-processing> [retrieved: January, 2013].
- [15] StreamBase, “StreamBase CEP.” [Online]. Available: <http://www.streambase.com> [retrieved: January, 2013].
- [16] O. Etzion and P. Niblett, *Event processing in action*. Manning Publications Co., 2010.
- [17] O. Papaemmanouil, U. Cetintemel, and J. Jannotti, “Integrating Pub-Sub and Stream Processing for Internet-Scale Monitoring,” 2009.
- [18] E. Fidler, H. A. Jacobsen, G. Li, and S. Mankovski, “The PADRES Distributed Publish/Subscribe System,” in *8th International Conference on Feature Interactions in Telecommunications and Software Systems*, 2005.
- [19] G. Baptista, M. Endler, and J. Viterbo, “Middleware Supporting Situational Awareness in Mission-Critical Scenarios with Rotorcraft,” PUC-Rio, Rio de Janeiro, 2012.
- [20] D. H. Mahlinh, *Coordinate Systems and Map Projections*. London: George Philip and Sons Publishers, 1973.
- [21] “FalconView.” [Online]. Available: <http://www.falconview.org/trac/FalconView> [retrieved: January, 2013].
- [22] Raytheon, “Solipsys Tactical Display Framework (TDF).” [Online]. Available: <http://www.solipsys.com/tdf/> [retrieved: January, 2013].

# A Methodology for Evaluating Complex Interactions between Multiple Autonomic Managers

Thaddeus Eze, Richard Anthony, Chris Walshaw, and Alan Soper  
 Autonomic Computing Research Group  
 School of Computing & Mathematical Sciences (CMS)  
 University of Greenwich, London, United Kingdom  
 {T.O.Eze, R.J.Anthony, C.Walshaw and A.J.Soper}@gre.ac.uk

**Abstract** — the very success of autonomic systems has inevitably led to situations where multiple autonomic managers need to coexist and/or interact directly or indirectly within the same system. This is evident, for example, in the increasing availability of large datacentres with multiple [heterogeneous] managers which are independently designed. Potentially, problems can arise as a result of *conflict-of-interest* when these managers (components) coexist. There is a growing concern that the lack of support for interoperability will become a break issue for future systems. We present an architecture-based solution to interoperability. Our approach is based on a Trustworthy Autonomic Architecture (different from traditional autonomic computing architecture) that includes mechanisms and instrumentation to explicitly support interoperability and trustworthiness. We posit that interoperability support should be designed in and integral at the architectural level, and not treated as add-ons as it cannot be reliably retro-fitted to systems. In this work-in-progress paper, we analyse the issue of interoperability and present our approach using a datacentre multi-manager scenario.

**Keywords**- *autonomic computing; interoperability; datacentre; multi-manager*

## I. INTRODUCTION

Autonomic Computing has progressively grown to become a mainstream concept. Earlier efforts were fundamentally concerned with getting autonomic computing to work and establishing fundamental concepts and demonstrating viability. Many mechanisms and techniques have been explored. Now that the concept of autonomic computing is well understood and widely accepted (and almost becoming commonplace), the focus has shifted to, amongst other things, addressing issues of scale and heterogeneity [1]. The increase in scale and size (of, e.g., datacentres) coupled with heterogeneity of services and platforms means that more Autonomic Managers (AMs) could be integrated to achieve a particular goal, e.g., datacentre optimisation. This has led to the need for interoperability between AMs. Interoperability deals with how to manage multi-manager scenarios, to govern complex interactions between managers and to arbitrate when conflicts arise. On the horizon these are the kind of challenges facing the autonomic computing research community [1][3].

The challenge of multi-manager interactions can be understandably enormous. This stems from the fact that, for example, components (and indeed AMs) could be multi-vendor supplied, upgrades in one manager could trigger

unfamiliar events, scalability can introduce bottlenecks, one manager may be unaware of the existence of another, and managers, though tested and perfected in isolation, may not have been wired at design to coexist with other managers. Multi-manager coexistence leads to potential conflicts. A typical example is illustrated with a multi-manager datacentre scenario: consider a datacentre with two independent AMs working together (unaware of each other) to optimise the use of the datacentre –a Performance Manager (P<sub>c</sub>M) optimises resource provisioning to maintain service level agreement (SLA). It does this by dynamically (re)allocating resources and maintaining a pool of idle servers to ensure high responsiveness to high priority applications. A Power Manager (P<sub>o</sub>M) seeks to optimise power usage (as power is one of the major cost overheads of datacentres [4]) by shutting down servers that have been idle for a certain length of time. Although each manager performs brilliantly in isolation but by coexisting, the success of one manager defeats the goal of another –one seeks to shutdown a server that another seeks to keep alive. The (in)activities of one manager affect the costs of provisioning (e.g., delay cost, scheduling cost, competition cost etc.) for another in one way or the other. One way of mitigating this conflict is to have an external agent that can detect and diagnose the problem. The problem with this is that it introduces more complexity (e.g., any AM addition will require rewiring of other AMs) as system is scaled up (adding complexity in the process of solving a complexity problem) which is not desirable.

We have in [5] proposed a Trustworthy Autonomic Architecture (TAA). The TAA architecture, presented in Section III, employs a nested control loop technique to explicitly support run-time validation, dependability and trustworthiness. The *DependabilityCheck* component of the TAA provides a way of logically arbitrating between coexisting AMs. We present our interoperability approach in Section IV and conclude the work in Section V.

## II. BACKGROUND

Kephart *et al* [2] presents a clear demonstration of the need for interoperability mechanisms. In that work two independently-developed AMs were implemented: the first dealt with application resource management (specifically CPU usage optimisation) and the second, a power manager, dealt with modulating the operating frequency of the CPU to ensure that the power cap was not exceeded. It was shown

that without a means to interact, both managers throttled and sped up the CPU without recourse to one another, thereby failing to achieve their intended optimisations and potentially destabilising the system. We envisage widespread repetition of this problem until a universally accepted approach to interoperability is implemented.

Richard *et al* [3] evaluates the nature and scope of the interoperability challenges for autonomic systems, identifies a set of requirements for a universal solution and proposes a service-based approach to interoperability to handle both direct and indirect conflicts in a multi-manager scenario. In this approach, an Interoperability Service (IS) interacts with autonomic managers through a dedicated interface and is able to detect possible conflicts of management interests. In this way the IS manages all interoperability activities by granting or withholding management rights to different autonomic managers as appropriate. [3] discusses two types of conflicts in a multi-manager scenario: Direct conflicts occur where AMs attempt to manage the same explicit resource while indirect conflicts arise when AMs control different resources, but the management effects of one have an undesirable impact on the management function of the other. This latter type of conflict, in our opinion, is the most frequent and problematic, as there are such a wide variety of unpredictable ways in which such conflicts can occur.

Other works focus on bespoke interoperability solution [6], direct AMs interactions at the level of autonomic elements to ensure that management obligations are met [7], hierarchical relationship to autonomic element interactions [8] and MAPE architecture modification [9] where it is suggested to separate out the Monitoring and Analysis stages of the MAPE loop into distinct autonomic elements, with designed-in interactions between them.

The research community has made valuable progress towards AM interoperability but this progress is yet to lead to a standardised approach. Although the current state of practice is a significant step, an equally significant issue is that they do not tackle the problem of unintended or unexpected interactions that can occur when independently developed AMs co-exist in a system [3]. Further from that, and more realistically, AMs may not need to know about the existence of other managers –they are designed in isolation (probably by different vendors) and operate differently (for different goals) without recourse to one another. So, to have close-coupled interoperability (i.e., where specific actions in one AM react to, or complement those of another), the source code and detailed functional specifications of each AM must be available to all AMs. This is near impossible and where possible, requires a rewiring of each AM whenever a new AM is added. These are why we look to the autonomic architecture to provide us a solution –hence, our architecture-based approach. We posit that to avoid introducing further complexity through solving the interoperability problem, the autonomic architecture should envision (and provide for) interoperability support from the scratch. This is to say that the autonomic architecture should be dynamic enough to accommodate expected and unexpected developments.

### III. THE TRUSTWORTHY AUTONOMIC ARCHITECTURE

TAA is an autonomic architectural framework that integrates three critical engine blocks (AC – *AutonomicController*, VC – *ValidationCheck* and DC – *DependabilityCheck*) in a modular fashion to lend autonomic systems extended (and robust) behavioural scope and trustability. These building blocks are implemented as modular components which are then connected to give the required trusted and dependable structure. To summarise the workings of TAA (see Figure 1), a system performs basic functions (to achieve its fundamental objectives) without any intelligent control of its activities. An autonomic manager (AC) is introduced to add some smartness by intelligently controlling the decision-making of the system. The actions of the manager are validated (VC) for correctness before they are actuated. A longer term control (DC) considers the behaviour of the manager over a period of time (after a certain number of decisions) to determine the effect of the manager’s intervention on the system and to take corrective action (arbitrate) if need be. VC and DC can inhibit the decisions or actions of AC. For complete details of TAA see [5].

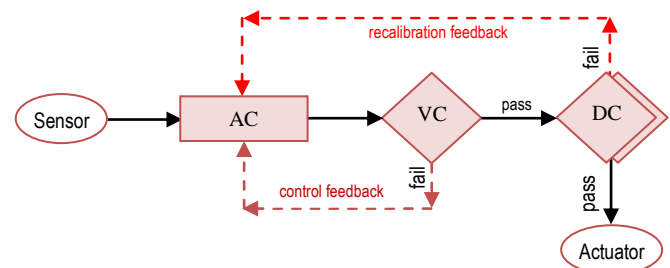


Figure 1: Detailed trustworthy autonomic architecture

In most of the autonomic systems, autonomic components are almost satisfactorily sufficient to provide required autonomic solution but in the longer term (e.g., as in multi-manager scenario), these rely on external supervision (typically by human) to extend their behavioural scope and trustability. This is resolved by the DC component. We rely on this component to address the interoperability problem as explained in Section IV. One of the powers of autonomic is its contextual generic implication and adaptation of terms and technologies. These are tailored to suit context and operational requirements. This quality allows us to adapt the TAA components (especially the DC) which can define, as necessary, stability and interoperability goals etc.

### IV. THE ARCHITECTURE-BASED INTEROPERABILITY

Let us consider, in more details (Figure 2), the multi-manager datacentre example presented earlier in Section I: the datacentre comprises a pool of resources  $S_i$  (live servers), a pool of shutdown servers  $\tilde{S}_i$  (ready to be powered and restored to  $S_i$  as need be), a list of applications  $A_j$ , a pool of services  $\mathcal{U}$  (a combination of applications and their provisioning servers), and two AMs (performance manager  $P_eM$  and a power manager  $P_oM$ ) that optimise the entire system.  $A_j$  and  $S_i$  are, respectively, a collection of applications supported (as services) by the datacentre and a collection of servers



available to the manager for provisioning available services according to request. As service requests arrive,  $P_eM$  dynamically populates  $\mathcal{U}$  to service the requests.  $\mathcal{U}$  is defined by:

$$U = \begin{cases} A_1: (S_{11}, S_{12}, S_{13}, \dots, S_{1i}) \\ A_2: (S_{21}, S_{22}, S_{23}, \dots, S_{2i}) \\ \dots \dots \dots \dots \dots \\ A_n: (S_{n1}, S_{n2}, S_{n3}, \dots, S_{ni}) \end{cases} \quad (1)$$

Where  $n$  is the number of application entries into  $\mathcal{U}$ . (1) indicates that a server can be (re)deployed for different applications. All the servers  $i$  in  $S_i$  are up and running (constantly available –or so desired by  $P_eM$ ) waiting for (re)deployment. The primary performance goal of  $P_eM$  is to minimise oscillation and maximise stability (including just-in-time service delivery) while the secondary performance goal is to maximise throughput. The goal of  $P_eM$ , on the other hand, is to optimise power consumption. This task is simply achieved by shutting down any server that has been idle for time  $T_s$ . Figure 2 details how TAA is used to manage interoperability between  $P_eM$  and  $P_oM$ .

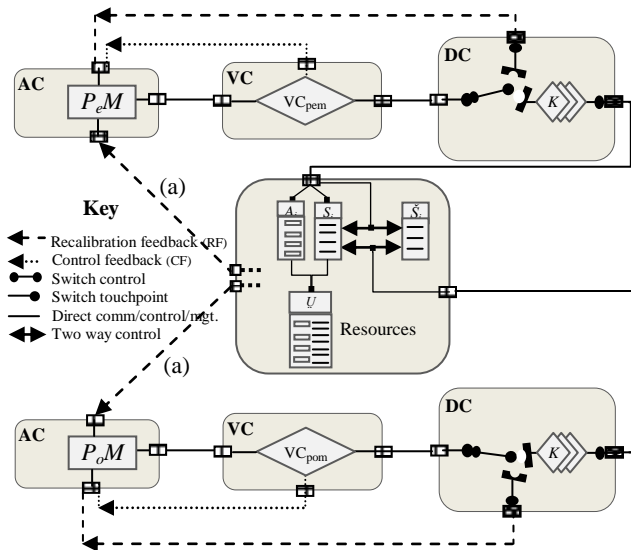


Figure 2: The DC component provides interoperability management

Figure 2 shows the communications and control within the components of the proposed architecture. The managers take performance decisions which are then validated by their respective VC ( $VC_{pom}$  and  $VC_{pem}$ ) for correctness. A CF is generated if validation fails and with this feedback, the manager adjusts its decisions. The DC takes a longer term view of the managers’ behaviour and either allows a manager to carry on with its actions (if check passes) or generates a RF otherwise. DC contains other subcomponents ( $K$ ), e.g., interoperability, stability etc. [1] but for brevity, we will concentrate on the interoperability subcomponent here.

The interoperability component is implemented using knowledge-based technology. It learns and keeps track of the system’s state following the passed decisions of the manager. If after a number of decision instances the manager senses a conflict with its decisions (based on expected versus actual

system state), another RF is generated (a) to retune the manager’s decisions. Take for instance, if after some time  $P_oM$  notices that the same set of servers it has shutdown have constantly come back live without it powering them, there is only one conclusion: another operation (probably a human, another manager, etc.) is not ‘happy’ with  $P_oM$ ’s decisions. So,  $P_oM$ ’s DC generates a RF with an appropriate tuning parameter value ( $\beta$ ) to throttle the size of  $T_s$  (2). By sensing the effects of its actions and dynamically throttling  $T_s$  within an acceptable boundary,  $P_oM$  is able to coexist with any other manager. Notice that the two managers do not need to know any details or even the existence of each other. In real life, this is typical of two staff that share an office space but work at different times. If both return next day and find the office rearranged, they will both adjust in their arrangement of the office until an accepted structure is reached. This can be achieved without both getting to meet. DC provides extra capacity for a manager to dynamically throttle its behaviour to suit the goal of the system.

$$T_s = (T_s \times \beta) \quad (2)$$

There are costs associated with the operations of a datacentre. These costs are affected in one way or the other by the actions of the managers. We identify three costs (Table I) which are used in our experiment –this is not exhaustive.

TABLE I: OPERATION COSTS

Cost	Description
Delay	Server booting and configuration time. Affects application performance
Scheduling	Reconfiguration and rescheduling time. Resource is unavailable during this time
Competition	One application has all resources and the other suffers

Apart from the costs mentioned in Table I above, other measurables from our experiment for analysing the performances of the managers include:

- Tracking SLA: service level will be measured as service delivery ratio (ratio of service delivery to service request) with an optimised value of 1.
  - o Values above 1 indicate over provisioning which comes at a cost
  - o Values below 1 indicate proximity to SLA
  - o Server provisioning can be throttled to track SLA
- Impact of the manager on the above metrics over time

Figure 3 is a front-end snapshot of the system (still under design) which models our multi-manager datacentre case scenario example and analyses the performances of the managers. The system allows for the simulation of three different scenarios of coexisting managers. This provides for three coexisting options for  $P_oM$  and  $P_eM$ : in the first option (with AC component), the managers operate autonomously without any interoperability support; the second option (with AC and VC components) introduces local run-time validation within individual manager and without any interoperability support; the third option (with AC, VC and DC components) introduces, amongst other controls, interoperability support.

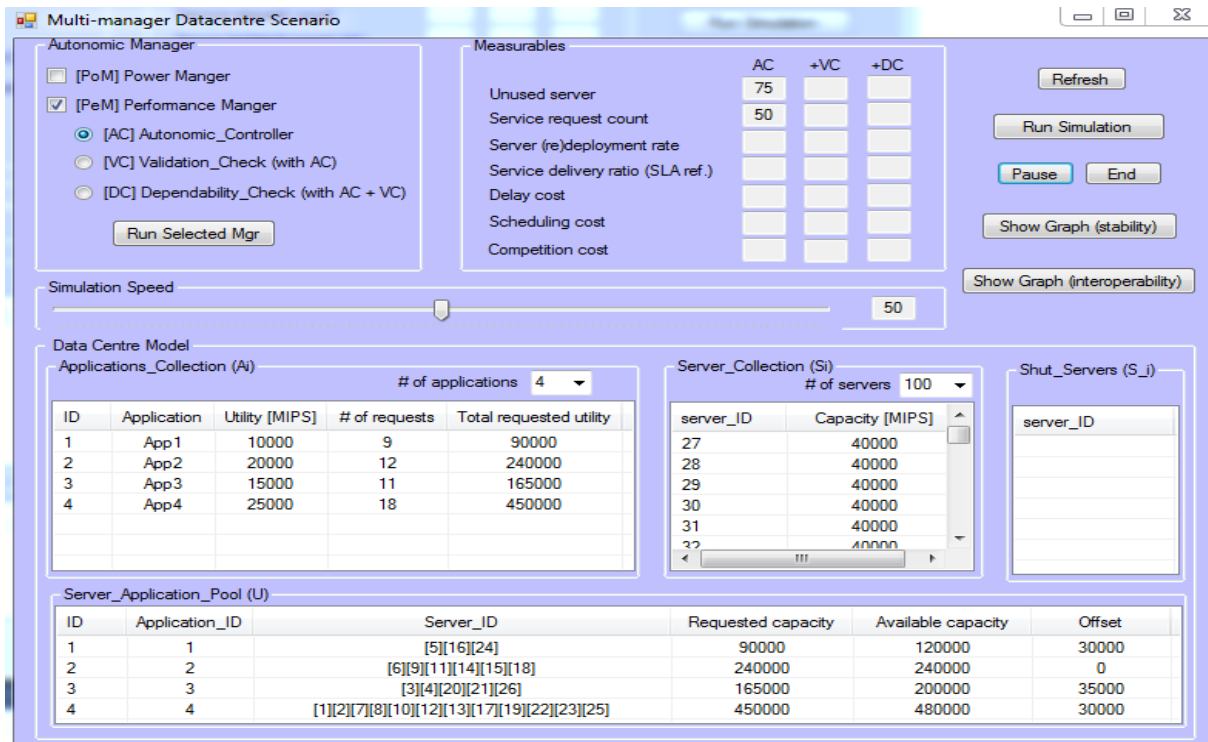


Figure 3: Multi-manager datacentre application

The third option is the main focus of this work. Other simulation options are also possible, e.g., selecting only PeM and running the above three options. On running the simulation, a script generates service requests. The service requests are measured in MIPS (million instructions per second). In the end, the performances of the managers (with and without interoperability support) are analysed against the listed measurables. This will identify, amongst other things, the effect/impact of our interoperability solution on the coexistence of the two managers.

V. CONCLUSION

We have presented, in this work-in-progress paper, an architecture-based interoperability solution. The solution is based on our earlier proposed trustworthy autonomic architecture. The architecture, which can be adapted to support several autonomic solutions, includes mechanisms and instrumentation to explicitly support run-time validation, interoperability and trustworthiness. We posit that to avoid introducing further complexity through solving the interoperability problem, the autonomic architecture should envision (and provide for) interoperability support from the scratch. This is to say that the autonomic architecture should be dynamic enough to accommodate expected and unexpected developments.

We analysed a multi-manager datacentre case example that represents a typical scenario of coexisting managers that leads to potential conflicts. This evaluates the nature and scope of the interoperability challenge and the need for a solution. We have also introduced an application that models the case example scenario. The next line of action is to run series of experiments once the case example application is

fully completed. Results, analysis and further details will be published subsequently.

REFERENCES

- [1] T. Eze, R. Anthony, C. Walshaw, and A. Soper, "Autonomic Computing in the First Decade: Trends and Direction," 8th Int'l Conference on Autonomic and Autonomous Systems (ICAS, St. Maarten 2012), pp. 80-85.
- [2] J. Kephart, H. Chan, R. Das, and D. Levine, "Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs," 4th Int'l Conference on Autonomic Computing (ICAC, Florida, USA, 2007).
- [3] R. Anthony, M. Pelc, and H. Shauib, "The Interoperability Challenge for Autonomic Computing," 3rd Int'l Conference on Emerging Network Intelligence (EMERGING, Lisbon, Portugal, 2011).
- [4] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang, "Power and Performance Management of Virtualized Computing Environments via Lookahead Control," 5th Int'l Conference on Autonomic Computing (ICAC, Illinois, USA, 2008).
- [5] T. Eze, R. Anthony, C. Walshaw, and A. Soper, "A New Architecture for Trustworthy Autonomic Systems," 4th Int'l Conference on Emerging Network Intelligence (EMERGING, Barcelona, Spain, 2012).
- [6] M. Wang, N. Kandasamy, A. Guezl, and M. Kam, "Adaptive performance control of computing systems via distributed cooperative control: Application to power management in computing clusters," 3rd Intl. Conf. on Autonomic Computing (ICAC, Dublin, Ireland, 2006).
- [7] M. Zhao, J. Xu, and J. Figueiredo, "Towards autonomic grid data management with virtualized distributed file systems" 3rd Int'l Conference on Autonomic Computing (ICAC, Dublin, Ireland, 2006).
- [8] B. Khargharia, S. Hariri, and S. Yousif, "Autonomic power and performance management for computing systems," 3rd Int'l Conference on Autonomic Computing (ICAC, Dublin, Ireland, 2006).
- [9] M. Kutare, G. Eisenhauer, C. Wang, K. Schwan, V. Talwar, and M. Wolf, "Monalytics: Online monitoring and analytics for managing large scale data centers," 7th Int'l Conference on Autonomic Computing (ICAC, Washington DC, USA, 2010).
- [10] R. Anthony, "Policy-based autonomic computing with integral support for self-stabilisation," Int. Journal of Autonomic Computing, Vol. 1, No. 1, 2009, pp. 1-33.

# Organic Programming of Real-Time Operating Systems

Lial Khaluf<sup>1</sup> and Franz Josef Rammig<sup>2</sup>

Heinz Nixdorf Institute

University of Paderborn

Paderborn, Germany

Email<sup>1</sup>: klial@hni.uni-paderborn.de

Email<sup>2</sup>: franz@upb.de

**Abstract**—Self-adaptability and self-management have become nowadays challenging properties of real-time operating systems. Since the evolution of these systems as a response to environmental changes is restricted due to real-time constraints, it becomes more difficult for the system to adapt itself and manage its stability at run time. This paper introduces a new approach, which applies the organic programming concept to real-time operating systems. The approach enables to define a self-adaptable and self-management real-time operating system.

*Keywords-cell; task; scheduling; real-time*

## I. INTRODUCTION

Real-time operating systems serve a set of tasks with respect to real time constraints. However, this set may grow and change over time. To allow this evolution, the real-time operating system must be able to adapt itself to the new circumstances and to manage its data in order to preserve all real time constraints. To achieve this goal, this paper introduces a new approach, which allows the tasks in a real-time application to behave like objects do in our real world. Objects in the real world can be adapted to serve a specific goal. Also creatures can change their behavior according to a set of influential factors. Similarly, there are a lot of situations encountered by real-time applications, where a modification of structure or behavior is needed (as a result of a task arrival or update). E.g., in a rescue system, if a robot task is defined to run toward a burning building using its wheels, and at run time it is decided by the system developer to get use of the wind factor. This change aims to fasten the robot movement or save some energy. In this case, the system must be able to modify the task structure (adding the wind factor) and the task behavior (consider the wind factor in the task functionality). In addition, if this task modification results in violating real-time constraints when an acceptance test is made for the updated task together with the other system tasks, then the system must have the ability to modify the structure or behavior of the other tasks at run time to increase the time capacity of the system. The new capacity may in turn allow the adaptation of the new changes with respect to real-time constraints. This ability is called organic programming [1]. In this paper, a new approach is provided to turn a real-time operating system (RTOS) into a self-adaptable and self-managing system. Here, the Organic Reconfigurable Operating System (ORCOS) [2] is used as an example for an underlying RTOS. The approach introduces the concept of real-time cells, and defines their structure and

behavior aiming at increasing the time capacity of the system for accepting new task arrivals and updates. The following section gives an overview of the related work. Section 3 describes the structure and behavior of real-time cells, and Section 4 shows the adapting algorithm of the system. Finally, a conclusion of the approach and a summary of possible future work are pointed out in Section 5.

## II. RELATED WORK

Self-adaptation and self-management are properties, which many approaches have aimed to realize, e.g., Ercatons in [1] have realized the concept of a true thing, which is able to adapt new changes at run time. However, this adaptation cannot be done under real-time constraints. In the real-time area, some systems [3][4] were developed to adapt themselves to a larger processing capacity at run time, but the used techniques may result in a large unbounded time overhead. The approach in [5], being by our knowledge one of the most recent and closest approaches to ours, has defined different profiles with different resource requirements for each task and allowed to choose the best combination of profiles at run time to adapt the system to certain situations. However, these profiles are developed offline, and new ones cannot be added to the systems at run time, which decreases the system adaptation ability. The approach in this paper applies the concept of organic programming on ORCOS to turn it into a self-adaptable and self-managing system. This is done by giving the ability to modify and develop the tasks online in a way that preserves all real-time constraints.

## III. STRUCTURE AND BEHAVIOR OF REAL-TIME CELLS

A real-time cell, referred to as RTC, is a system component mapped to a piece of memory to which a task is assigned. There are two kinds of RTCs, controlling RTCs and controlled RTCs. The first kind cannot change its behavior at run time and it is assigned a task before the system starts running. The second kind can change its structure and behavior at run time, and it is assigned a task online after the system starts running. At the moment, the presented approach is restricted only on periodic tasks scheduled by EDF [6]. However, the approach may be extended later to support in addition aperiodic tasks using an appropriate server like Total Bandwidth Server [7]. A task is usually defined by its data and functionality. The task in the proposed approach is similar to the usual task model but with an additional set of meta data.

The meta data differs between the tasks that can be assigned to controlling cells, called controlling tasks, and tasks that can be assigned to controlled cells, called controlled tasks. The meta data of both controlling and controlled tasks is used by the functionality of controlling tasks to be able to modify the controlled RTCs at run time. A controlled task is assumed to exist by mean of various versions (similar to profiles in [5]), called members in this paper to avoid confusion with the classical versioning model. All members of a task can accomplish the same basic functionality. However, the characteristics in various dimensions (memory requirements, precision, power dissipation, execution time,...) can vary. For simplicity reasons in this paper, it is assumed that all objectives other than execution time can be expressed by a single parameter, called *Cost*, so a cell is characterized by (cost, execution time). A new member may be added from the outside at any time. The adaptation process described in Section 4 is affecting controlled task members, which are currently in the system.

An RTC is active when its current task instance is executing (i.e., ready or running) and is non-active otherwise. The approach here defines only one controlling RTC, called the Engine-RTC. Its controlling task, called the Engine-Task, is responsible for accepting a new RTC or changing the structure and behavior of already existing controlled RTCs at run time. In this context, a controlled RTC is changed by changing its associated task. The new controlled task should accomplish the same basic functionality, which was expected from the old one. Whenever replacing a task member as part of the dynamic change test (see Section 4), this should also increase the time capacity of the system sufficiently, e.g., it may include functions or procedures with time characteristics different from the older ones. To allow this, other parameters usually have to be modified as well, e.g., the precision of calculations may be reduced. In other words, the system may have the ability to adapt itself to any new circumstances through choosing possible alternatives of the currently executing controlled tasks. As there might be a variety of choices that provide sufficient additional processor capacity, the goal is to find the solution, which came with the minimal costs concerning all other parameters. The meta data of each controlled task consists of the following information:

- **ID/Member**

$$ID/Member; id \in \{1, 2, \dots\}$$

$$\text{and } Member \in \{0, 1, 2, \dots\}$$

*ID* is a unique number to differentiate between the controlled tasks. *Member* is a number to point to a controlled task alternative. *ID/Member* is read by the Engine-RTC whenever a new controlled task arrives to realize if it is a new task (its *ID* does not exist in the system) or an update of an existing controlled task (its *ID* exists in the system, but the *Member* number does not exist for this *ID*).

- **CriticalityDegree**

$$CriticalityDegree \in \{1, 2, \dots\}$$

*CriticalityDegree* is a number to express how critical a controlled task is. It has the same value for all members of this task. Criticality increases whenever the *CriticalityDegree* decreases and vice versa.

- **UpdatingPoints**

$$UpdatingPoints \in \{(i, RelativeTimePoint - i);$$

$$i \in \{0, 1, 2, \dots\}\}$$

*UpdatingPoints* are certain points in the functionality code of the controlled task where replacing this task with another member of it is possible. Each updating point is defined with a number *i*, and the time *RelativeTimePoint - i* at which the replacement can take place. E.g., if the execution time equals 10 time units, and the controlled task has two updating points (0,0),(1,9), this means that the controlled task can be replaced before it starts executing or after 9 time units. The replacement of a controlled task at some updating point means modifying the controlled RTC to which this controlled task is assigned (switching to another task member). The release time of a next instance of a periodic task constitutes a natural updating point. In this paper, the adapting algorithm is first dedicated only for natural updating points and then refined for intermediate updating points.

- **Location**

$$Location \in \{(x, y);$$

$$x \in \{ComputeNode_0, ComputeNode_1, \dots\}$$

$$\text{and } y \in \{MemoryAddress_0, MemoryAddress_1, \dots\}$$

*Location* is defined by *ComputeNode<sub>i</sub>*, the compute node on which the controlled task resides, and *MemoryAddress<sub>i</sub>*, its memory address on that specific compute node. *Location* is used by the Engine-RTC to fetch the controlled task from its compute node to the compute node of ORCOS (*ComputeNode<sub>0</sub>*).

- **ArrivalTime** is the time at which the current instance of the controlled task should start executing. *ArrivalTime* might not exist in the meta data of a controlled task, in which case, ORCOS is informed that this task might be needed in the future for replacing a certain controlled task for which it is an alternative.
- **DeadlineTime** is the period of time within which the controlled task execution should be completed.
- **FetchingTime** is the time required to fetch a controlled task from the compute node where it resides to the compute node where ORCOS resides.
- **Dependency**

$$Dependency \in \{true, false\}$$

*Dependency* is true when the controlled task depends on the completion of other controlled tasks execution, otherwise *Dependency* is false. For simplicity, the adapting algorithm in this paper assumes an independent task set, which means that *Dependency* is false.

- **RelatedTasks**

$RelatedTasks \in \{(id_0/M_0, L_0, FT_0), (id_1/M_1, L_1, FT_1), \dots\};$   
 $id_i/M_i \in \{ID/Member\}; L_i \in Location$   
 and  $FT_i$  is the  
 FetchingTime of the controlled task $i\}$

*RelatedTasks* determines the identity, location and fetching time of the controlled tasks on which the execution of the controlled task depends. *RelatedTasks* is used by the Engine-RTC to get information about these tasks, since the new controlled task is only accepted in the system if its *RelatedTasks* can also be accepted. Please note that for simplicity reasons within this paper, the set of *RelatedTasks* is assumed to be empty.

- **ExecutionTime** = time required to execute the controlled task + *ExecutionTime* of the *RelatedTasks* + time required to register the *UpdatingPoints* in the meta data of the Engine-RTC whenever an updating point is reached at run time.

The previous definition assumes that all *RelatedTasks* have not started execution before the controlled task does. The *ExecutionTime* of any controlled task is bounded since all its factors are bounded. The reason is that the time required to execute any controlled task and the number of the *UpdatingPoints* for any controlled task are defined offline.

The Engine-Task is responsible for:

- registering the meta data of newly arrived controlled tasks or newly arrived members.
- making an acceptance test at new arrivals to the set of controlled tasks, the new arrival, and the Engine-task with respect to its worst case execution time. The approach here assumes the EDF scheduling algorithm, where all tasks are periodic and their period is equal to their *DeadlineTime*. Thus the acceptance test is  $\sum_{i=1}^n C_i/T_i \leq 1$ , where  $n$  is the number of tasks,  $C_i$  is the execution time, and  $T_i$  is the *DeadlineTime*.
- making a dynamic change test. This test is made in case the new arrival cannot be accepted to the currently existing system (a set of controlled tasks). The test verifies if changing the structure and behavior of RTCs according to certain rules (see the next section) can enable the acceptance of the new arrival.
- making a dynamic change, which means changing the structure and behavior of RTCs selected by the previous test. This includes fetching the controlled tasks, which are going to replace the controlled task of the selected RTCs from their compute nodes. In addition, it includes fetching the new controlled task or the new upgrading controlled task member, for which the dynamic change test is done.
- assigning the new controlled task or the new upgrading member (if it is accepted) to a controlled RTC.

- updating the meta data of the Engine-Task. This is important to evaluate the worst case execution time of the Engine-Task.

The meta data of the Engine-Task consists of the following:

- **ID=0**, differentiates the Engine-Task from other tasks in the system.
- **Location**

$Location \in \{(ComputeNode_0, MemoryAddress_i); i \in \{0, 1, 2, \dots\}\}$

*Location* defines the memory address of the Engine-Task on the compute node where ORCOS resides.

- **DeadlineTime** is the period of time within which the execution of the current instance of the Engine-Task has to be completed. This time is updated whenever the *WorstCaseExecutionTime* of the Engine-Task is updated.
- **TaskArray**

$TaskArray = \{v_{i,j}; v_{i,j}$  is a member $_i$  of a controlled task $_j\}$

*TaskArray* is an array dedicated to store controlled tasks members. Each column represents a controlled task, and includes elements representing the registered members of this controlled task. Each element includes the meta data of the member. Each column also includes three additional elements, the first one holds the last updating point registered by the controlled task, which is represented by this column. The second one indicates the number of elements currently stored for this task and the third one holds the identity *ID/Member* of the currently executed member of the controlled task represented by this column.

- **NumberOfTasks** equals the number of columns in *TaskArray*.
- **WorstCaseExecutionTime (WCET)** = registration time + acceptance test time + worst case dynamic change test time + worst case dynamic change time + worst case assigning time + time for updating the meta data.

The *WorstCaseExecutionTime* is bounded since all its factors are bounded, even those, which depend on the number of controlled tasks and the number of their members. These numbers are updated with each execution of the Engine-Task, and since the *WCET* is calculated after each execution of the Engine-Task, this means that the numbers are updated and the *WCET* of these factors can be predicted.

#### IV. THE ADAPTING ALGORITHM OF THE SYSTEM

The approach assumes EDF to be the scheduling algorithm used. Whenever the Engine-RTC becomes active, the Engine-Task is assigned the highest priority. This is done by assigning a reserved fraction of the processor capacity to the Engine-Task, where this fraction can vary over time, but is always known. At the start of the system, the Engine-RTC is non-active. Whenever a new controlled task or a new member of

a controlled task is added to a compute node, the node sends the meta data of this task to ORCOS. If the Engine-RTC is non-active, a system call is made to make it active. If the meta data includes an arrival time, two cases are to be considered:

1) Arrival of a new controlled task:

The Engine-task performs an acceptance test  $\sum_{i=1}^n C_i/T_i \leq 1$  on all tasks including the newly arriving one;

$C_i = ExecutionTime$  of the  $i$ th task.

$T_i = DeadlineTime$  of the  $i$ th task.

The Engine-task with its *WCET* is taken into consideration when making the acceptance test to ensure enough time capacity for its execution when a new arrival happens after this one. I.e., the adaptation of the processor bandwidth dedicated to the Engine-Task is considered as well.

If this arrival can be accepted, the Engine-Task fetches the new controlled Task and assigns it to a Controlled RTC. Otherwise, a dynamic change test must be run. The goal is to replace some of the current controlled tasks with alternative members, so that the acceptance test succeeds. In other words, the Engine-task searches for the controlled tasks members of minimal *Cost*, which can substitute the current utilizations  $C_i/T_i$  with smaller ones. Thus, the dynamic change test is executed as follows:

Let the columns of the *TaskArray* have the search order starting from the column, which represents the controlled task with the smallest remaining execution time to the controlled task with biggest remaining execution time. I.e., we want to provide the closest possible acceptance time. If several tasks have the same remaining execution time the task with highest *CriticalityDegree* is to be looked at first. For the next column  $j$  in the *TaskArray*{

- a) Let  $V_j$  be the currently executing controlled task member of the column  $j$ .
- b) Let  $F_j = C_j/T_j$  be the processor utilization for  $V_j$ .
- c) For each member  $V_{i,j}$ {
  - calculate the utilization  $F_{i,j}$ :  
 $F_{i,j} = C_{i,j}/T_{i,j}$ ; where  
 $C_{i,j}$  is the *Execution-Time* of  $V_{i,j}$   
 $T_{i,j}$  is the *Deadline-Time* of  $V_{i,j}$
  - $Gain_{i,j} = F_j - F_{i,j}$
  - If  $Gain_{i,j}$  is positive, extract the respective member's cost:  $Cost_{i,j}$
  - Add  $V_{i,j}$  with  $(Gain_{i,j}, Cost_{i,j})$  to a set of candidate members.}
- d) Select the subset of candidate members of minimal accumulated cost whose accumulated gain is sufficient to accept the new task.
- e) If the accumulated gain is sufficient: break; else inspect next task until all tasks have been visited.}

If the new task set is not schedulable, i.e., the maximal accumulated gain is not sufficient, then refuse the newly arrived controlled task.

Otherwise, replace every selected  $V_j$  by the respective selected  $V_{i,j}$ .

2) Arrival of a new update member of a controlled task:

The same previous procedure is followed, and the new update starts executing at the next natural updating point of the current executing instance of the controlled task.

The adapting algorithm could be refined to consider updating points other than natural ones. In this case, the previous two cases follow the same steps with one difference. The columns of the *TaskArray* must have the search order starting from the column, which represents the controlled task with the smallest execution time remaining to reach the next updating point to the controlled task with biggest execution time remaining to reach the next updating point. If two tasks have the same execution time remaining to reach the next updating point, the task with the highest *CriticalityDegree* is to be searched first.

## V. CONCLUSION AND FUTURE WORK

This paper has introduced an approach to develop a self-adapted and self-management RTOS. This was done by applying the concept of organic programming on RTOS (ORCOS in this case). Further work in this area could be done by, e.g., generalizing the approach to be applied to a larger set of scheduling algorithms, or considering other factors that may have influence on the time capacity, e.g., the non critical tasks, the communication between tasks, memory and other kinds of resources. Currently, we are pursuing to make an experimental evaluation of the principle approach on the RTOS ORCOS.

## REFERENCES

- [1] F. Langhammer, O. Imbusch, and G. von Walter, "Ercatons and Organic Programming: Say Good-Bye to Planned Economy," Organic Computing - Controlled Emergence, Vol. 06031, 2006.
- [2] <<https://orcos.cs.uni-paderborn.de/orcos>>, [retrieved: January, 2013].
- [3] J. A. Stankovic and K. Ramamritham, "The Spring Kernel: A New Paradigm for Real Time Operating Systems," Operating Systems Review (SIGOPS), Vol. 23, no. 3, February.1989, pp. 54-71.
- [4] K. Ramamritham and J. A. Stankovic, "Scheduling Algorithms and Operating Systems Support for Real Time Systems," Proceedings of the IEEE, Vol. 82, no. 1, January.1994, pp. 55-67.
- [5] S. Oberthür, L. Zaramba, and H. Lichte, "Flexible Resource Management for Self-X Systems: An Evaluation," in Proceedings of ISORCW'10, May.2010, pp. 1-10.
- [6] W. A. Horn, "Some Simple Scheduling Algorithms," Naval Research Logistics Quaterly, Vol. 21, no. 1, March.1974, pp. 177-185.
- [7] G. C. Butazzo, "Hard Real-Time Computing Systems. Predictable Scheduling Algorithms and Applications," Springer Science+Business Media, LLC, 2nd rev.ED, 2004.

## Higher Order Sliding Mode Control of Robot Manipulator

Awatif Guendouzi

Laboratory Robotics and Artificial  
Intelligence,  
CDTA, Algiers, Algeria  
guendouzi.awatif@gmail.com

Ahcene Boubakir

LAMEL, Faculty of Science and  
Technology,  
University of Jijel, Jijel, Algeria  
ah\_boubakir@yahoo.fr

Mustapha Hamerlain

Laboratory Robotics and Artificial  
Intelligence,  
CDTA, Algiers, Algeria  
mhamerlain@cdta.dz

**Abstract**—This paper presents the development of a nonlinear control strategy for a robot manipulator model, using a robust higher order sliding mode control structure. In the present work, a traditional sliding mode control is presented, the robustness of the controller in the context of stabilization and trajectory tracking, is analytically proved using Lyapunov approach. In order to reduce the chattering in sliding mode controller (SMC) we used the higher order sliding mode control algorithm (Super twisting and Twisting). The simulation results presented in this paper indicate that the suggested approach has considerable advantages compared to the classical sliding mode control.

**Keywords**-robot manipulator; higher order sliding mode control; Twisting; Super twisting

### I. INTRODUCTION

Variable structure systems with a sliding mode were discussed first in the Soviet literature [1], and have been widely developed in recent years. The sliding mode control (SMC) is a powerful method to control high-order nonlinear dynamic systems operating under uncertainty conditions [2][3]. A SMC law is designed such that the representative points' trajectories of the closed-loop system are attracted to the sliding surface and once on the sliding surface they slide towards the origin. As the sliding surface is hit, the system response is governed by the surface dynamic; consequently, the robustness to the uncertainty or disturbance is achieved. In spite of claimed robustness properties, high frequency oscillations of the state trajectories around the sliding manifold known as chattering phenomenon [2][4] are the major obstacles for the implementation of SMC in a wide range of applications.

Several methods of chattering reduction have been reported [5][6]. One approach [7] places a boundary layer around the switching surface such that the relay control is replaced by a saturation function. Another method higher order SMC [8][9][10], latter approach have been proposed for a Flexible Robot Arm in [11] [12]. The current papers result is based on this latter approach and its main idea can be described as follows:

Let  $s(x, t)$  ( $x \in \mathfrak{R}^n$  is the state variable,  $t \in \mathfrak{R}^+$  the time variable) be the sliding variable and  $r \in N$  the sliding order. The control forces to zero in finite time  $s$  and its  $(r-1)$  first higher time derivatives by acting discontinuously on the  $r$ th time derivative of  $s$ . Keeping the main advantages

of standard SMC, the chattering effect is eliminated and higher order precision is provided. In the case of "real" SMC [9], if  $\tau$  is the sampling time, the error is  $o(\tau)$  in the case of standard SMC [13] whereas it is  $o(\tau^r)$  in the  $r$ th order SMC [14].

In the case of second order SMC ( $r=2$ ), many works have given solutions. Several second order sliding mode algorithms are proposed in [9][14][15] [16].

The present paper proposes a multi-input multi-output (MIMO) second order sliding mode strategy, for this purpose we have chosen as an application PUMA 560 robot manipulator with three degrees of freedom model obtained using Lagrangian's equations [17]. The proposed controller based on sliding mode control approach.

The paper is arranged as follows: Section 2 introduces a general PUM 560 robot manipulator model. Section 3 presents traditional sliding mode controller design. Section 4 displays the design of the second order SMC (the Twisting and de Super Twisting algorithm). Section 5 presents the simulation results obtained with the full dynamic model. Finally, we present the comparative study to show the effectiveness and feasibility of the proposed control strategy.

### II. DYNAMIC MODEL OF ROBOT MANIPULATOR

To control the manipulator arms, we chose the model of industrial PUMA 560 robot manipulator presented in Figure 1. We considered only the first three rotational joints  $q_1, q_2$  and  $q_3$ . The dynamic model [17] is given in simplified matrix form in (1).

$$M(q)\ddot{q} + V_m(q, \dot{q})\dot{q} + G(q) + u_{m_0} = u \quad (1)$$

with:

$q \in \mathfrak{R}^n$  : Vector of joint positions;

$\dot{q} \in \mathfrak{R}^n$  : Vector of joint velocities;

$\ddot{q} \in \mathfrak{R}^n$  : Vector of joint accelerations;

$u \in \mathfrak{R}^n$  : Vector of forces and / or torques of motor;

$$u = [u_1, u_2, u_3]^T \quad (2)$$

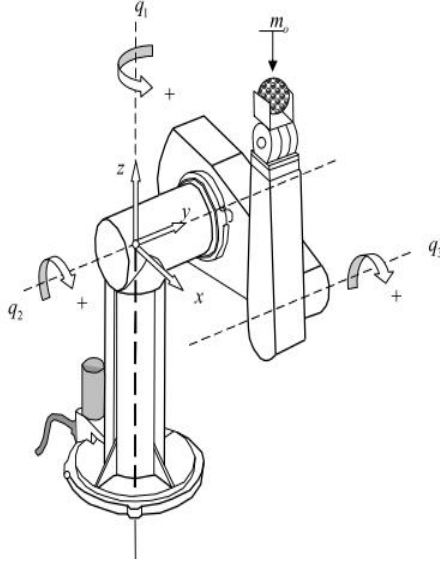


Figure 1. PUMA 560 robot manipulator

$u_{m_o}(t) \in \mathfrak{R}^3$  : Vector of torque due to the load  $m_o$ .

$$u_{m_o} = m_o J^T(q) [J(q)\ddot{q} + \dot{J}(q, \dot{q})\dot{q} + g] \quad (3)$$

where the Jacobian matrix is defined by:

$$J(q) = \begin{bmatrix} -s_1(l_2c_2 + l_3c_{23}) - d_2c_1 & -c_1(l_2s_2 + l_3s_{23}) & -c_1(l_3s_{23}) \\ c_1(l_2c_2 + l_3c_{23}) - d_2s_1 & -s_1(l_2s_2 + l_3s_{23}) & -s_1(l_3s_{23}) \\ 0 & -(l_2c_2 + l_3c_{23}) & -(l_3c_{23}) \end{bmatrix} \quad (4)$$

$\dot{J}(q, \dot{q})$  derived from the Jacobian matrix obtained from the differentiation with respect to time.

$M(q) \in \mathfrak{R}^{n \times n}$  : Symmetric positive definite matrix of inertial accelerations;

$$M(q) = \begin{bmatrix} I_1 + I_2c_2^2 + I_3c_2^2 + I_4c_2c_{23} & I_5s_{23} + I_6s_2 & I_5s_{23} \\ I_5s_{23} + I_6s_2 & I_7 + I_4c_3 & I_8 + 0.5I_4c_3 \\ I_5s_{23} & I_8 + 0.5I_4c_3 & I_9 \end{bmatrix} \quad (5)$$

$G(q) \in \mathfrak{R}^n$  : Vector of forces and / or couples due to gravitational forces;

$$G(q) = \begin{bmatrix} 0 \\ -(m_3l_2 + 0.5m_2l_2)gc_2 - 0.5m_3l_3gc_{23} \\ -0.5m_3l_3gc_{23} \end{bmatrix} \quad (7)$$

$V_m(q, \dot{q}) \in \mathfrak{R}^{n \times n}$  : Matrix of forces and / or torques due to centrifugal and Coriolis accelerations;

$$V_m(q, \dot{q}) \cdot \dot{q} = \begin{bmatrix} -(2(I_3s_2c_2 + I_2s_2c_2c_{23}) + I_4(c_2s_{23} + s_2c_{23}))\dot{q}_1\dot{q}_2 \\ -(2I_2s_2c_2c_{23} + I_4c_2s_{23})\dot{q}_1\dot{q}_3 \\ +(I_6c_2 + I_5c_2c_{23})\dot{q}_2^2 + (2I_5c_2c_{23})\dot{q}_2\dot{q}_3 + (I_5c_2c_{23})\dot{q}_3^2 \\ (I_3c_2s_2 + I_2c_2c_2s_{23} + 0.5I_4(s_2c_2c_{23} + c_2s_{23}))\dot{q}_1^2 \\ -(I_4s_3)\dot{q}_2\dot{q}_3 - (0.5I_4s_3)\dot{q}_3^2 \\ (I_2s_2c_2c_{23} + 0.5I_4c_2s_{23})\dot{q}_1^2 + (0.5I_4s_3)\dot{q}_2^2 \end{bmatrix} \quad (6)$$

with the following notations:

$$\begin{cases} c_i = \cos(q_i), & c_{ij} = \cos(q_i + q_j), \\ s_i = \sin(q_i), & s_{ij} = \sin(q_i + q_j), \end{cases} \quad (8)$$

1) *Property 1*: The matrix  $M(q)$  is symmetric, positive definite and bounded, and its inverse is existing and also bounded. The matrix verifies the following equality  $\dot{M}(q) - 2V_m(q, \dot{q})$  for none a zero vector  $X$  :

$$X^T [\dot{M}(q) - 2V_m(q, \dot{q})]X = 0 \quad (9)$$

### III. SLIDING MODE CONTROLLER DESIGN

This section focuses on the design of a sliding mode control for the stabilization of nonlinear systems; we will apply it on a highly nonlinear system which is the PUMA 560 robot manipulator. This control must meet the specifications defining the objectives, including stability, speed, accuracy and robustness. The simulations are performed in the case of trajectory tracking, and we passed the drop test load as the test of robustness. The PUMA 560 robot model without taking into account the effect of the load is as follows:

$$M\ddot{q} + V_m\dot{q} + G = u \quad (10)$$

This model describes the dynamics of a robot manipulator with three degrees of freedom, which requires the synthesis of three controls, and as each joint is considered as a subsystem whose relative degree is  $r_i = 2$ , which means that each surface  $s_i$  is of order  $r_i - 1$ . And the error  $e$  is defined by:  $e = q - q_d$ , with  $e_i = q_i - q_{id}$

Therefore, the sliding surface is chosen,  $s = [s_1, s_2, s_3]$ , such as:

$$s_i = \dot{e}_i + \lambda_i e_i \quad (11)$$

we can write,  $s = \dot{q} - (\dot{q}_d - \lambda e)$



where,

$$s = \dot{q} - \dot{q}_r \quad (12)$$

with,  $\dot{q}_r = (\dot{q}_d - \lambda e)$  is considered a reference for the joint velocity.

From (10) and (12), we can set:

$$\begin{aligned} M \cdot \dot{s} &= u - V_m \cdot \dot{q} - G - M \cdot \ddot{q}_r \\ M \cdot \dot{s} &= u - V_m \cdot (s + \dot{q}_r) - G - M \cdot \ddot{q}_r \\ M \cdot \dot{s} &= u - M \cdot \ddot{q}_r - V_m \cdot s - V_m \cdot \dot{q}_r - G \end{aligned} \quad (13)$$

#### A. Proposition 1

We define the Lyapunov function as follows:

$$V = \frac{1}{2} s^T M s \quad (14)$$

From property (1), the matrix  $M$  is positive definite, and we also  $V > 0$  for  $s \neq 0$ . So, according to the study of Lyapunov stability, the system is stable when,  $\dot{V} < 0$  with  $\dot{V}$  is the time derivative of  $V$  such that:

$$\begin{cases} \dot{V} = s^T M \dot{s} + \frac{1}{2} s^T \dot{M} s \\ \dot{V} = s^T [u - M \ddot{q}_r - V_m s - V_m \dot{q}_r - G] + \frac{1}{2} s^T \dot{M} s \end{cases} \quad (15)$$

and from (9), we have  $s^T [\dot{M} - 2V_m] s = 0$ .

This implies that:

$$\dot{V} = s^T [u - M \ddot{q}_r - V_m \dot{q}_r - G] \quad (16)$$

#### B. Proposition 2

The control signal  $u$ , be given by

$$u = u_{eq} - K^T \text{sign}(s) \quad (17)$$

and

$$u_{eq} = M \ddot{q}_r + V_m \dot{q}_r + G \quad (18)$$

with the gains of switching  $K = [K_1, K_2, K_2]^T$ ,  $K_i > 0$ .

#### C. Proof 1

Thus, with this choice of the control  $u$  from (17), we obtain:

$$\dot{V} = -s^T K^T \text{sign}(s) = -|s| K^T < 0 \quad (19)$$

Therefore, with the control (17) the system is stable in closed loop (the equilibrium point  $e_i = 0$ , with  $i = 1, 3$ , is asymptotically stable).

#### IV. 2-SLIDING MODE CONTROLLER DESIGN

The problem of 2-sliding mode control is to constrain the trajectories of the system to evolve on the sliding manifold in a finite time [15]:

$$s_2 = \{x \in O : s(t, x) = \dot{s}(t, x) = 0\} \quad (20)$$

Let us write again the dynamic model of PUMA 560 robot manipulator as follows:

$$M \ddot{q} + V_m \dot{q} + G = u \quad (21)$$

For each joint, the constraint is chosen linear function:

$$s_i = \dot{e}_i + \lambda_i e_i, \quad \lambda_i > 0, \quad i = 1:3 \quad (22)$$

The equivalent control is given by:

$$u_{eq} = M \ddot{q}_r + V_m \dot{q}_r + G \quad (23)$$

For each subsystem, the effective control  $u$  is composed of two terms: the equivalent control  $u_{eq}$  and the discontinuous control.

To calculate the latter, we use the algorithms of 2-sliding mode control. When we take local coordinates  $[y_1 \ y_2]^T = [s \ \dot{s}]^T$ , the problem of 2-sliding mode for each subsystem (joint) is reduced to the stabilization in finite time of the auxiliary system of order two below:

$$\begin{cases} \dot{y}_1 = y_2 \\ \dot{y}_2 = \chi(t, x) + \zeta(t, x) \nu \end{cases} \quad (24)$$

where  $\nu$  represents the derivative of the control by respect to time  $\nu = \dot{u}$ ; in this case, the control  $u$  is considered a state variable.

$$\begin{cases} \chi(x, t) = \frac{\partial}{\partial t} \dot{s}(t, x, u) + \frac{\partial}{\partial x} \dot{s}(t, x, u) \{f(x) + g(x)u\} \\ \zeta(t, x) = \frac{\partial}{\partial u} \dot{s}(t, x, u) \end{cases} \quad (25)$$

$0 < K_m \leq \left| \frac{\partial \dot{s}}{\partial u} \right| \leq K_M$ , with the necessary condition of existence of the equivalent control in sliding mode  $\frac{\partial \dot{s}}{\partial u} \neq 0$  and  $\left| \frac{\partial}{\partial t} \dot{s}(t, x, u) + \frac{\partial}{\partial x} \dot{s}(t, x, u) \{f(x) + g(x)u\} \right| \leq \psi$ .

Two 2-sliding algorithms (the twisting and Super Twisting) are applied to demonstrate their ability to stabilize the system, reduce chattering and improve accuracy in a problem tracking.

#### A. Twisting Algorithm Controller Design

The control law for a system of relative degree  $r = 1$  is as follows in (26) [15]:

$$v = \dot{u} = \begin{cases} -u & \text{if } |u| > u_M \\ -a_m \text{sign}(y_1) & \text{if } y_1 y_2 \leq 0 \text{ and } |u| \leq u_M \\ -a_M \text{sign}(y_1) & \text{if } y_1 y_2 > 0 \text{ and } |u| \leq u_M \end{cases} \quad (26)$$

with sufficient conditions for finite time convergence are:

$$a_M > 4 \frac{K_m}{\varepsilon_0}, a_m > \frac{\Psi}{K_m}, K_m a_M - \Psi > K_M a_m + \Psi \quad (27)$$

The effective control is as follows:  $u = u_{eq} + \int \dot{u} dt$ , we note

$$u_{TW} = \int \dot{u} dt.$$

In practice for an real sliding, instead of  $y_2$ , determination of the sign of  $y_1 y_2$  is made by the first difference of  $y_1$  as:

$$\Delta s = \begin{cases} 0 & \text{for } K = 0 \\ (y_1(K\tau) - y_1((K-1)\tau)) & \text{for } K \geq 1 \end{cases} \quad (28)$$

where  $\tau$  is the sampling period.

#### B. Super Twisting Algorithm Controller Design

In this section, we will apply the super Twisting algorithm [15] to stabilize the robot manipulator. The effective control  $u$  for this algorithm consists of two terms:

$$u_{eq} \text{ and } u_{ST}, \quad u(t) = u_{eq}(t) + u_{ST}(t) \quad \text{with}$$

$$u_{ST}(t) = \int \dot{u}_1(t) dt + u_2(t).$$

and

$$\dot{u}_1(t) = \begin{cases} -u & \text{if } |u| > u_M \\ -W \text{sign}(y_1) & \text{if } |u| \leq u_M \end{cases} \quad (29)$$

$$u_2(t) = \begin{cases} -\lambda \varepsilon_0^\rho \text{sign}(y_1) & \text{if } y_1 > \varepsilon_0 \\ -\lambda |y_1|^\rho \text{sign}(y_1) & \text{if } y_1 \leq \varepsilon_0 \end{cases} \quad (30)$$

In this case, sufficient conditions for convergence are:

$$W > \frac{\Psi}{K_m}, \lambda^2 \geq 4\psi \frac{K_M}{K_m^2} \left( \frac{W + \Psi}{W - \Psi} \right), 0 < \rho \leq 0.5 \quad (31)$$

## V. SIMULATION RESULTS

In this section, simulations are presented to illustrate the performance and robustness of proposed control law when applied to PUMA 560 robot manipulator. The parameters values used for the dynamic model are as follows [17],

$$M(q)\ddot{q} + V_m(q, \dot{q})\dot{q} + G(q) + u_{m_0} = u \quad (32)$$

- Mass of various links  
 $m_2 = 17.40 \text{ kg}$ ,  $m_3 = 5.04 \text{ kg}$ ,  $m_4 = 0.82 \text{ kg}$ ,  
 $m_5 = 0.35 \text{ kg}$ ,  $m_6 = 0.09 \text{ kg}$ ,  
 $m_l = m_4 + m_5 + m_6 = 1.26 \text{ kg}$ .
- Geometrical parameters  
 $d_2 = 149.09 \text{ mm}$ ,  $l_2 = 431.8 \text{ mm}$ ,  $l_3 = 433.07 \text{ mm}$ .

$$\text{with } g = [0 \ 0 \ 9.8]^T$$

To excite the any dynamics of robot there is a cycloidal trajectory test (33), where the different joints move respectively of the position  $\{-50^\circ, -135^\circ, 135^\circ\}$  to the position  $\{45^\circ, -85^\circ, 30^\circ\}$ , in a time of movement equal to 1.5 sec.

$$q_{di}(t) = \begin{cases} q_{di}(0) + \frac{D_i}{2\pi} \left[ 2\pi \frac{t}{t_f} - \sin\left(2\pi \frac{t}{t_f}\right) \right] & \text{for } 0 \leq t \leq t_f \\ q_{di}(t_f) & \text{for } t_f < t \end{cases} \quad (33)$$

with  $D_i = q_{di}(t_f) - q_{di}(0)$ : Displacement, and  $t_f$  the final instant of the movement.

The parameters of the simulations are,

- For the sliding surfaces  $\lambda_j = 5$ ,  $j = 1, 2, 3$
- For a SMC control, the control gains selected are  $K^T = [50, 50, 50]$
- For the Twisting algorithm are:  $a_m^T = [0.1, 0.1, 0.1]$ ,  $a_M^T = [550, 550, 550]$ ,  $u_M^T = [20, 20, 20]$ .
- For the Super Twisting algorithm are:  $\varepsilon_0 = 0.1$ ,  $\rho = 0.5$ ,  $u_M^T = [20, 20, 20]$ ,  $\lambda^T = [250, 250, 250]$ ,  $W^T = [0.025, 0.025, 0.025]$ .

with the sampling period  $T = 0.001 \text{ sec}$ , and the simulation time  $t = 2 \text{ sec}$ . To test the robustness of the control laws proposed, we chose the falling load. In this, the robot is a tracking trajectory with a load ( $m_0 \text{ kg}$ ) and if the maximum speed is pending, the load drops ( $m_0 = 0 \text{ à } t = 0.8$ ).

## VI. COMPARATIVE STUDY

The results of using conventional sliding mode controller are shown in Figure 2 and the results obtained

when the 2-sliding controller has been used are shown in Figure 3 for Twisting algorithm, Figure 4 for Super Twisting algorithm.

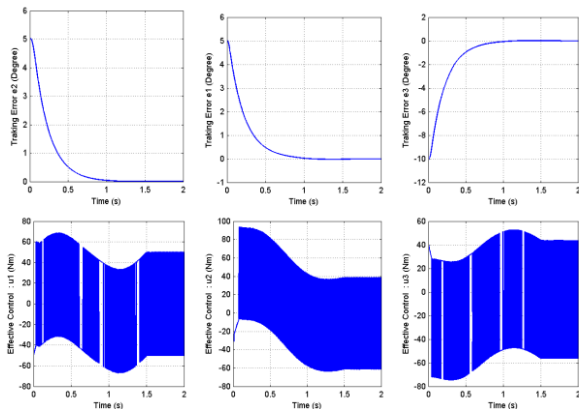


Figure 2. Simulation results of first-order SMC.

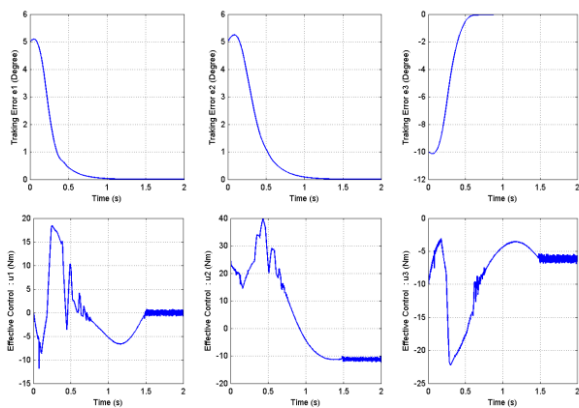


Figure 3. Simulation results of second order SMC: Twisting algorithm.

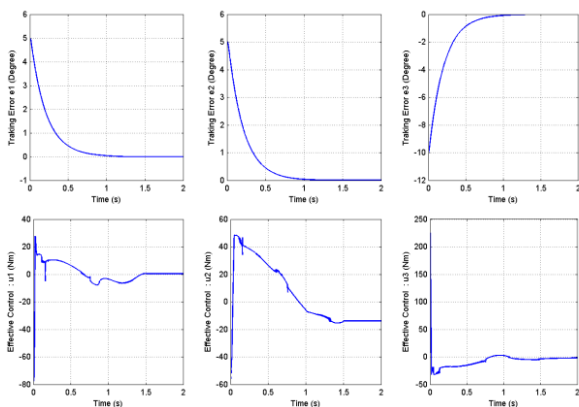


Figure 4. Simulation results of second order SMC: Super Twisting algorithm.

From the simulations results, we can find that the control result of conventional sliding mode controller produces a serious chattering phenomenon, Figure 2.

On the other hand, the chattering phenomenon of the controlled system is suppressed in the case of Super Twisting algorithm controller (Figure 4). Moreover, the

proposed controller is a robust controller since the load drops hasn't influence on the control performances.

To compare the performance of 2-sliding controller with SMC, we define two cost functions  $J_1$  and  $J_2$ , such as:

$$J_1 = \frac{1}{2} \cdot \sum_{k=1}^P (u^T u) \tag{34}$$

$$J_2 = \frac{1}{2} \cdot \sum_{k=1}^P (e^T e) \tag{35}$$

The simulation results for each performance index are given in Table I:

TABLE I. COMPARATIVE STUDY

Controller	Cost function $J_1$	Cost function $J_2$
SMC	6.6686 $10^6$	2.6692
Twisting algorithm	3.9922 $10^5$	28.0488
Super Twisting algorithm	2.769 $10^5$	2.8437

Comparing the simulation results, it can be said that the proposed control strategy, second order SMC (Super Twisting algorithm), gave better performance than using the conventional sliding mode controller.

## VII. CONCLUSION AND FUTURE WORK

In this paper, a second order SMC is proposed for a PUMA 560 robot manipulator system and simulation results are presented. Firstly, the classical sliding mode control of PUMA 560 robot manipulator system is developed. Secondly, the second order SMC control is used to smooth the discontinuous control term in order to alleviate the chattering phenomenon. The simulation results presented in this paper indicate that the suggested approach has considerable advantages compared to the classical sliding mode control. As future works, we would like making an experimental study of the control approach on a real robot manipulator.

## REFERENCES

- [1] S. V. Emel'yanov, "Variable Structure Control Systems," Moscow: Nauka, 1967.
- [2] V. I. Utkin, "Variable structure systems with sliding modes," IEEE Transactions on Automatic Control, vol. 26, 1977, pp. 212-222.
- [3] V. I. Utkin, "Sliding mode in control and optimization," Berlin: Springer, 1992.
- [4] M. Hamerlain, "An Anthropomorphic Robot Arm Driven by Artificial Muscles using a Variable Structure control," IROS'95, IEEE/RSJ International Conference On Intelligent Robots and systems, Pittsburgh, Pennsylvania, USA, August 5-9, 1995, pp. 550- 555.
- [5] M. Chettouh, R. Toumi, and M. Hamerlain, "Chatter reduction in an artificial muscles robot application," International Journal of Robotics and Automation, vol. 23, no. 2, 2008, pp. 88-97.

- [6] M. Hamerlain, T. Youssef, and M. Belhocine, "Switching on the derivative of control for decreasing the chattering," *IEE Journal on Control Theory and Application*, vol. 148, Issue 1, 2001, pp. 88-96.
- [7] J.J. Slotine and S.S. Sastry, "tracking control of nonlinear systems using sliding surfaces with application to robot manipulators," *International Journal of Control*, vol. 38, 1983, pp.465-492.
- [8] G. Bartolini, A. Ferrara, and E. Usai, "Chattering avoidance by second order sliding mode control," *IEEE Transactions on Automatic Control*, vol. 43, 1998, pp. 241-246.
- [9] A. Levant, A. Pridor, R. Gitizadeh, I. Yaesh, and J. Z Ben-Asher, "Aircraft pitch control via second-order sliding technique". *AIAA Journal of Guidance, Control and Dynamics*, vol. 23, no. 4, 2000, pp. 586-594.
- [10] A. Levant, "Universal siso sliding-mode controllers with finite-time convergence," *IEEE Transactions on Automatic Control*, vol. 49, 2001, pp. 1447-1451.
- [11] M. Chettouh, R. Toumi, and M. Hamerlain, "High order sliding modes for a robot driven by pneumatic artificial rubber muscles," *Advanced Robotics*, vol. 22, no. 6, 2008, pp. 689-704.
- [12] K. Braikia, M. Chettouh, B. Tondu, P. Acco, and M. Hamerlain, "Improved Control Strategy of 2-Sliding Controls Applied to a Flexible Robot Arm," *Advanced Robotics*, vol. 25, 2011, pp. 1515-1538.
- [13] K. Furuta, "Sliding mode control of discrete system," *Systems and Control Letters*, vol. 14, 1990, pp.145-152.
- [14] G. Bartolini, A. Ferrara, E. Usai, and V. I. Utkin, "On multi-input chattering-free second-order sliding mode control," *IEEE Transactions on Automatic Control*, vol. 45, 2000, pp. 1711-1717.
- [15] A. Levant, "Construction principles of 2-sliding mode design," *Automatica*, vol. 43, 2007, pp. 576-586.
- [16] A. Rezoug, M. Hamerlain, and M. Tadjine, "Adaptive Interval Type-2 Fuzzy Controller for Robot Arm Actuated By Pneumatic Artificial Rubber Muscles", *The Mediterranean Journal of Measurement and Control*, vol. 7, no. 3, August 2011.
- [17] S. Alavandar and M.J. Nigam, "Fuzzy PD+I control of a six DOF robot manipulator", *Industrial Robot: An International Journal*, vol. 35 Iss: 2, 2008, pp. 125-132.

## Numerical Model of Generalized Multi-Priority Queuing System

Eimutis Valakevicius, Mindaugas Snipas  
 Department of Mathematical Research in Systems  
 Kaunas University of Technology  
 Kaunas, Lithuania  
 E-mails: {eimval@ktu.lt, minsnip@ktu.lt}

**Abstract**—The paper presents a novel approach of semi-automatic technique for creation numerical models of stochastic queuing systems of general type. The general model can be reduced to a queuing system of desired construction: with various numbers of queues, servers, any service time distributions and different priorities. Markov chains are used to model the dynamics of systems with phase-type distributed service times. Event-driven approach is applied to model the change of the system. The calculated stationary probabilities of possible states of the system are used to compute measures of the system performance.

**Keywords**—generalized queueing system; numerical model; phase-type distribution; Markov chain; stationary probabilities

### I. INTRODUCTION

Queuing models are useful mathematical tools in the study of design problems of a wide variety of stochastic service systems, such as, computer and telecommunication networks, inventory, logistic and other complex systems. There are a lot of publications in scientific literature devoting to the construction of analytical models of Markovian queuing systems [1][2][3][4][5]. Some queuing systems are modeled using numerical approach. Usually, the analytic approach is effective only for Markovian queuing systems with infinite waiting room. However, it is difficult or often impossible to find analytical solutions of non-Markovian multi-class and multi-server priority queues. Some researches restricting themselves analyzing two priority classes or limiting the number of high or low priority jobs [6][7]. In other papers multi-priority, multi-server system is roughly approximated by a single server system [8] or aggregating the multi-priority classes into two classes, as in [9][10]. This article proposes a numerical approach for semi-automatic construction numerical models of complex generalized queuing systems. The numerical technique allows modeling a wide variety of stochastic service systems. The paper is an enhancement of articles published in [11][12] and can be considered as a “work in progress”. So, it was not possibility to compare the proposed technique with related work.

The paper is organized as follows. Section 2 gives description of a generalized queuing system under consideration. Section 3 describes the algorithm implemented in to programming tool to create a numerical model. The example is given in Section 4 and the paper is concluded in Section 5.

### II. DESCRIPTION OF THE STOCHASTIC MODEL

Consider a queuing system with  $k$  identical servers and  $n$  priority classes: the highest priority 1 and lowest priority  $n$ . Priority rules is non - preemptive. In queuing applications, it is often convenient to approximate the service time by distributions that are built out of a finite sum or a finite mixture of exponentially distributed components. These distributions are called phase-type distributions. The application of such distribution allows constructing Markovian model [13] of the system. The queuing system under consideration assumes Poisson arrival processes [14] of  $n$  class’s customers with rates  $\lambda_i, i = 1, \dots, n$  and phase-type distributed service times with parameters  $\mu_i^{(j)}, i = 1, \dots, k; j = 1, 2$ . Two phase-type distribution will be used in the model. The approximation of general type distribution by two phase-type distribution is discussed in [12]. The scheme of simplified queuing system with  $n = 2$  and  $k = 1$  is represented in Figure 1:

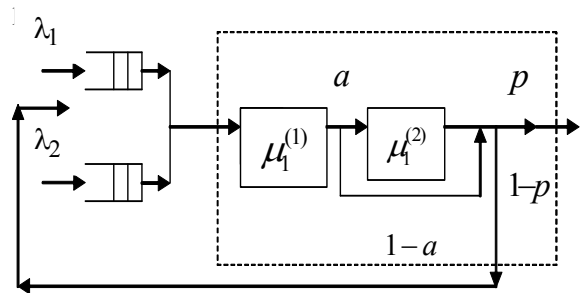


Figure 1. The simplified queuing system

Within each class, service discipline of customers is a First Come First Served (FCFS). Queue length of each class customers is limited. Each served customer can leave the system with probability  $p$  or can be returned for repeated service with probability  $1-p$ .

III. PROCESS OF CREATING A NUMERICAL MODEL OF THE QUEUEING SYSTEM

We will present the algorithm of creating the numerical model of a queueing system under consideration. It can be considered as consisting of five parts [15].

1. Define the state vector of the system. All possible combinations of coordinates describe the set of all possible system states.
2. Identify all the possible events which change the state of the system.
3. Describe all the events for generation system states and transition matrix.
4. Compute the stationary probabilities of the states.
5. Calculate the performance measures of the system.

The performance of the generalized queueing system will be described according the suggested algorithm.

The state of the system is given by the vector

$$S = (s_i, i = 1, \dots, n; s_j, j = n + 1, \dots, n + k; s_m, m = n + k + 1, \dots, n + 2k), \quad (1)$$

here

$s_i, i = \overline{1, n}$  - the number of customers in the  $i$ th queue.

$$s_j = \begin{cases} i, & \text{if the } (j-n) \text{ th server is busy} \\ & \text{by customer from } i \text{ th queue;} \\ 0, & \text{if the server is idle;} \\ & j = n + 1, \dots, n + k; \end{cases} \quad (2)$$

$$s_m = \begin{cases} 0, & \text{if the } (m-n-k) \text{ th server is idle;} \\ 1, & \text{if the customer is served} \\ & \text{in the } (m-n-k) \text{ th server in} \\ & \text{the first phase;} \\ 2, & \text{if the customer is served} \\ & \text{in the } (m-n-k) \text{ th server in the} \\ & \text{second phase;} \\ & m = n + k + 1, \dots, n + 2k; \end{cases} \quad (3)$$

The values of coordinates are bounded so:

$$s_i \leq l_i, i = 1, \dots, n. \quad (4)$$

The set of transition rates among states is the following:

$$Rates = \{ \lambda_i, i = 1, \dots, n, \mu_j^{(1)}, j = 1, \dots, k, \mu_j^{(2)}, j = 1, \dots, k \}, \quad (5)$$

where

$\lambda_i$  - the arrival rate of  $i$ th class customers to the system;

$\mu_j^{(i)}$  - the service rate of a customer in  $j$ th server and  $i$ th stage.

The set of possible events in the system:

$$E = \{ e_1; e_i, i = 2, \dots, n; e_j, j = n + 1, \dots, n + k; e_m, m = n + k + 1, \dots, n + 2k; e_r, r = n + 2k + 1, \dots, n + 3k; e_v, v = n + 3k + 1, \dots, n + 4k; e_t, t = n + 4k + 1, \dots, n + 5k \}, \quad (6)$$

where

$e_i$  - a customer of the  $i$ th class arrived to system with rate  $\lambda_i$ ;

$e_j$  - a customer was served in  $(j-n)$ th server in the first phase and with rate  $\mu_j^{(1)} p (1-a)$  leaves the system;

$e_m$  - a customer was served in  $(m-n-k)$ th server in the first phase and with rate  $\mu_{m-n-k}^{(1)} (1-p)(1-a)$  returns for repeated service;

$e_r$  - a customer was transferred from the first phase to the second phase in the  $(r-n-2k)$ th server with rate  $\mu_{r-n-k}^{(1)} a$ ;

$e_v$  - a customer was served in  $(v-n-3k)$  th server and left the system with rate  $\mu_{v-n-k}^{(2)} p$ ;

$e_t$  - a customer was served in  $(t-n-4k)$ th server and was returned for repeated service with rate  $\mu_{t-n-k}^{(2)} (1-p)$ ;

The initial state is

$$S_0 = (0, \dots, 0).$$

The description of the event  $e_1$  is given below.

$e_1$  :

```

    if  $s_1 < l_1$ 
    then if
    ( $s_1 > 0$  and ( $s_m = 1$  or  $s_m = 2$ )),
     $m = n + k + 1, \dots, n + 2k$ 
    then  $s_1 := s_1 + 1$ 
    end if
    else
     $j =: \min\{p : s_p = 0, p = n + 1, \dots, n + k\}$ 
     $s_j := 1, s_{j+k} := 1$ 
    end if
    Return Rate :=  $\lambda_1$ 
    
```

The created programming tool using described events generates the possible set of states, constructs the transition matrix among them, calculates stationary probabilities of system states

$$\pi(s_i, i = 1, \dots, n; s_j, j = n + 1, \dots, n + k; s_m, m = n + k + 1, \dots, n + 2k) := \pi(s_i; s_j; s_m) \quad (7)$$

and computes performance measures of the system. The formulas for computing desired measures must be created. For example, the mean queue length  $E(L_i)$  of  $i$ th class customers in the system is calculated according the following formula

$$E(L_i) = \sum_{s_i=1, \dots, n} \sum_{s_j, j=n+1, \dots, n+k} \sum_{s_m, m=n+k+1, \dots, n+2k} s_i \cdot \pi(s_i; s_j; s_m) \quad (8)$$

#### IV. EXAMPLE

Consider a queuing system with 3 priority classes and 2 identical servers. This queuing system assumes Poisson arrival processes with rates  $\lambda_i, i = 1, \dots, n$  and exponentially distributed service times with parameters  $\mu_j^{(1)}$  and  $\mu_j^{(2)} = 0, j = 1, \dots, k$  respectively.

The main problem when modelling real systems is a rapid growth of the number of states of a Markov chain. Computation of stationary probabilities can require a large amount of calculations and computer resources. For example, if the set of states is described as

$$S = \{(s_1, s_2, s_3, s_4, s_5)\}, s_1 \leq l_1; s_2 \leq l_2; s_3 \leq l_3; s_4 \leq l_4; s_5 \leq l_5, \quad (9)$$

it is easy to prove that the total number of states equal to

$$(l_1 + 1)(l_2 + 1)(l_3 + 1)(l_4 + 1)(l_5 + 1). \quad (10)$$

We chose limitation on summary waiting space

$$l_1 + l_2 + l_3 \leq 7. \quad (11)$$

We estimated the mean queue length for each customer class and general loss probability  $P(Loss)$  (i.e., the probability that the customer of any class will not be served).

Numerical modeling results were compared with simulation results using ARENA simulation software. During each simulation session total amount of more than 5 500 000 customer arrivals were generated. We used PC with AMD Athlon 64 X2 dual core processor 4000+ 2.10 GHz, 896 MB of RAM physical address extension. Intensity rates, modeling results and calculation times are in Table 1.

TABLE I MODELING RESULTS

Parameters	Method	$E(L_1)$	$E(L_2)$	$E(L_3)$	$P(Loss)$	Time, s.
$\lambda_1 = 1 \quad \lambda_2 = 0.8 \quad \lambda_3 = 0.5$	Numerical modeling	0.0722	0.0922	0.0800	0.0012	78
$\mu_1^{(1)} = 2 \quad \mu_2^{(1)} = 3 \quad \mu_3^{(1)} = 4$	ARENA	0.0719	0.0925	0.0795	0.0012	316
$\lambda_1 = 2 \quad \lambda_2 = 1.6 \quad \lambda_3 = 1$	Numerical modeling	0.4108	0.7774	1.0709	0.0764	79
$\mu_1^{(1)} = 2 \quad \mu_2^{(1)} = 3 \quad \mu_3^{(1)} = 4$	ARENA	0.4097	0.7783	1.0684	0.0763	314

The modeling parameters were chosen freely. The obtained results showed that it requires less calculation time and gives higher accuracy than standard simulation software to model certain queuing systems.

#### V. CONCLUSION AND FUTURE WORK

This paper presented a new programming tool, which was developed in C++ language code to implement a numerical model. The tool realized the semi-automatic technique for creation of numerical models and analysis of stochastic queuing systems. It generates the set of all possible states and transition matrix of the system under consideration, computes the stationary probabilities and the performance measures of the system according to the given formulas. The technique offers advanced functionality compared to existing tools from several points of view: 1) the tool allows solving problems of large scale (with thousands or millions of states); 2) the tool allows creating models of non-markovian queuing systems approximating ones by markovian models.

Problems of large dimension require a large amount of calculations and computer resources. Often it is impossible to solve some problems with available resources. The future work will be devoted for creation a special computation algorithm which allows avoiding mentioned problems.

#### REFERENCES

- [1] A. Sleptchenko, A. Harten, and M. Heijden, "An exact solution for the state probabilities of the multi-class, multi-server queue with preemptive priorities, *Queueing systems*", *Queueing Systems: Theory and Applications*, vol. 50, 2005, pp. 81-107.
- [2] H. R. Gail, S. L. Hantler, and B. A. Taylor, "On preemptive Markovian queue with multiple servers and two priority classes", *Mathematics of Operations research*, vol. 17, no. 2, 1992, pp. 365-391.
- [3] A. Sleptchenko, I. J. B. F. Adan, and G. J. van Houtum, "Joint queue length distribution of multi-class, single-server queues with preemptive priorities" <<http://alexandria.tue.nl/repository/books/581962.pdf>> 25.12.2012
- [4] M. Harchol-Balter, T. Osogami, A. Scheller-Wolf, and A. Wierman, "Multi-server queueing systems with multiple priority classes *Queueing Systems*", *Theory and Applications*, vol. 51, 2005, pp. 331-360.
- [5] E. P. C. Kao and S. D. Wilson, "Analysis of non-preemptive priority queues with multiple servers and two priority classes", *European Journal of Operational Research*, vol. 118, 1999, pp. 181-193.
- [6] E. P. C. Kao and K. S. Narayanan, "Computing steady-state probabilities of a nonpreemptive priority multiserver queue", *Journal on Computing*, vol. 2, no 3, 1990, pp. 211 – 218.
- [7] E. Kao and K. Narayanan, "Modeling a multiprocessor system with preemptive priorities", *Management Science*, vol. 2, 1991, pp.185–97.
- [8] A. Bondi and J. Buzen, "The response times of priority classes under preemptive resume in M/G/m queues", *In ACM Sigmetrics*, Aug. 1984, pp. 195–201.
- [9] I. Mitrani and P. King, "Multiprocessor systems with preemptive priorities", *Performance Evaluation*, vol.1, 1981, pp. 118–125.
- [10] T. Nishida, "Approximate analysis for heterogeneous multiprocessor systems with priority jobs", *Performance Evaluation*, vol. 15, 1992, pp. 77–88.
- [11] M. Snipas and E. Valakevicius, "Numerical-analytic model of multi-class, multi-server queue with nonpreemptive priorities", *Innovations and Advances in Computer Sciences and Engineering*. Dordrecht: Springer, 2010, ISBN 9789048136575, pp. 413-415.
- [12] E. Valakevicius, "The application of phase type distributions for modelling queuing systems", *Journal of Systemics, Cybernetics and Informatics*, vol. 5, no. 6, 2009, pp. 28-32.
- [13] W-K. Ching and M. K. Ng, "Markov Chains: Models, Algorithms and Applications", Springer, 2005.
- [14] H. C. Tijms, *Stochastic Models: An Algorithmic Approach*. John Wiley & Sons, 1994.
- [15] H. Pranevicius and E. Valakevicius, *Numerical Models of Systems Specified by Markovian processes*. Kaunas: Technologija, 1996.



# HiSPADA: Self-Organising Hierarchies for Large-Scale Multi-Agent Systems

Jan-Philipp Steghöfer, Pascal Behrmann, Gerrit Anders, Florian Siefert, and Wolfgang Reif  
 Institute for Software & Systems Engineering, University of Augsburg, Germany  
 {steghoefer, anders, siefert, reif}@informatik.uni-augsburg.de, PascalBehrmann@gmx.de

**Abstract**—The formation of hierarchies within large-scale systems can solve problems of scalability and distributed control. In this paper, we suggest a self-organising partitioning control scheme that uses a distributed set partitioning algorithm to dynamically introduce and resolve hierarchy layers in a decentralised fashion according to the needs of the application at runtime. The partitioning control can work within a predefined organisational framework and is highly adaptable to application-specific needs. We demonstrate the approach with an application from the domain of distributed power management and provide evaluations that show that a self-organising hierarchy formation can increase scalability by simplifying control decisions with negligible overhead.

**Keywords**—Autonomous agents; Hierarchical Systems; Adaptive Systems

## I. INTRODUCTION

Complex systems, consisting of thousands of individual, heterogeneous agents, are a scalability nightmare. Centralised approaches to their control and monitoring can almost never be realised. This is due to two facts: first of all, propagating the necessary data from the agents to a central instance can take a very long time; second, the decision making process has to consider all this data and therefore can have prohibiting runtime. While waiting for the data to be aggregated and for the decision to be made, the environment of the agents can change tremendously. It can change so much that the decision that is ultimately made becomes obsolete immediately.

For this reason, many systems (as outlined in Section II) have levels of indirection and abstraction, often represented by hierarchies [1]. Using hierarchies allows to compartmentalise computation to certain areas of the system and thus to provide scalability beyond a couple of dozen agents. Information aggregation does not take as long and decision making processes do not have to deal with such a high number of variables. Additionally, hierarchies can be used to represent existing organisational structures, a feature often desired when modelling existing systems with agents. Systems of systems (SoS) and holarchies are concepts that directly integrate this hierarchical nature. More often than not, however, hierarchies are defined once by the designers of the system and do not adapt to changing circumstances during the runtime of the system.

In this paper, we present *HiSPADA*, an extension of the Set Partitioning Algorithm for Distributed Agents (SPADA, [2]). *HiSPADA* forms hierarchical partitions that represent levels

in the system structure. These partitions and the hierarchy in which they are arranged are highly flexible and can adapt to changes in the environment, to new and leaving agents, to changing system goals, etc. We show that *HiSPADA* indeed reduces the complexity of control decisions while the system's other quality attributes remain almost the same.

However, to make use of a hierarchical partitioning control, some caveats have to be heeded. On the one hand, for *HiSPADA* to work efficiently, it must be possible to introduce new layers by creating intermediary agents that encapsulate the essence of the agents they control. On the other hand, hierarchical task decomposition must be possible. Our illustrative example exhibits these properties. The case study is a power management system in which distributed energy resources (DERs) are partitioned into Autonomous Virtual Power Plants (AVPPs) [3]. These AVPPs coordinate the power plants to meet power demands and to stabilise the network frequency. The system consisting of hierarchical AVPPs and DERs can be considered a system of systems.

To meet the overall power demand, schedules are created that assign a fraction of the demand to each individual power plant by solving a constraint optimisation problem. The more power plants there are in the system, the more complex the calculation of their schedules and the longer the runtime of the scheduling algorithms. AVPPs compartmentalise this complexity by reducing the number of power plants being part of the scheduling process within each individual AVPP. To find a good partitioning of power plants into AVPPs, the time required for scheduling has to be part of the decisions in the AVPP formation process. Additionally, existing utilities and transmission grid infrastructure impose given organisational structure within which it is possible to form AVPPs. The structure changes when AVPPs can not fulfil their power demand or other obligations.

This paper is structured as follows: in Section II, we introduce SoS and holarchies, illustrate systems in which predefined hierarchies are used for the same purpose as in this paper, introduce hierarchical clustering, and outline the ideas behind dynamical hierarchies. Section III then describes the non-hierarchical self-organisation algorithm that forms the basis of *HiSPADA*, which itself is explained in detail in Section IV along with its prerequisites. We present an evaluation of the algorithm in Section V and conclude the paper with a discussion and outlook on future work in Section VI.

## II. HOLARCHIES, HIERARCHICAL SELF-ORGANISATION, AND SYSTEMS OF SYSTEMS

In [1], Horling and Lesser discuss a number of organisational paradigms for multi-agent systems. Most importantly, they identify hierarchies and holarchies as the main approaches to deal with complexity and scalability issues.

Holarchies are a special kind of hierarchical organisation. The concept of a “holon”, originally described by Koestler in [4], refers to a recursive, self-similar structure, while a holarchy is a hierarchy of self-organised holons. In modern uses of this concept, e.g., for holonic manufacturing systems, a holarchy is defined as a system of such holons cooperating to achieve a common goal [5]. A holarchy used in this context is usually not changed at runtime but predefined by the designer to meet the specific system requirements.

Similarly, many self-organising systems use predefined, static hierarchies to reflect existing hierarchical structures. In the Organic Traffic Control [6] project, e.g., hierarchies consist of individual traffic lights, intersections, and entire roads. On each level, the system is able to learn traffic patterns and therefore adapts to the traffic flow. The different levels allow recognition of patterns on different scales and optimise, e.g., to create green waves during rush hour.

Hierarchical clustering algorithms are, e.g., regarded in sensor networks. There, groups of sensors represented by a cluster head are formed. The head serves as a communication hub for the entire group. The number of hops (nodes a message has to go through to reach its destination) and the communication range are vital decision variables. As [7] shows, hierarchies can significantly reduce the complexity of cluster formation. This result shows that hierarchies increase scalability. However, results and algorithms from sensor networks can not be readily applied to other domains as they work under specific assumptions. Conversely, it will be difficult to adapt HiSPADA to sensor networks as it does not account for energy efficiency or limited communication.

Holarchies and hierarchies are directly related to systems of systems (SoS). SoS are composed of systems that are themselves complex systems. They are usually distributed in nature and very large [8]. A lot of work on SoS originates in a military context (see, e.g., [9]) where the interconnection of different, complex systems is a must to provide battlefield information and control of a wide array of weapon systems and sensors. Although these systems are heavily connected, they remain independent in many ways. Key characteristics of SoS thus include functional and administrative independence of sub-systems and geographic distribution [10]. Furthermore, the behaviour of SoS is often emergent and its development evolutionary. This definition also applies to the case study used here and other open, heterogeneous systems.

Hierarchies also play a crucial role in the emergence of new functionality. The artificial life community has been looking into the synthesis of dynamical hierarchies [11] that

show different emergent behaviour at various scales. Another research direction are hierarchies that self-assemble without an explicit algorithm [12] such as the one presented here. While these approaches are very interesting, their usefulness for large-scale technical systems has not yet been proven.

## III. SET PARTITIONING WITH SPADA

SPADA [2], the Set Partitioning Algorithm for Distributed Agents, solves the so-called set partitioning problem (SPP) in a general, decentralised manner. In the SPP, the goal is to partition a set  $\mathcal{A} = \{a_1, \dots, a_n\}$  into  $k \leq n$  pairwise disjoint subsets, i.e., partitions, that exhibit application-specific properties. For example, if the objective is to group similar or dissimilar elements together, the SPP is equivalent to clustering or anticlustering [13]. This can be achieved by complementing the SPP with an appropriate metric. In case such a metric defines how well agents can work together on a common task, the SPP is equivalent to coalition structure generation [14]. Since SPADA has been designed to solve the SPP in general, it can be applied to these specific problems as well. This distinguishes SPADA from other centralised and decentralised approaches, which are often specialised to a specific problem in a specific domain. In the following, we give a short summary of SPADA’s basic functionality and characteristics. A more detailed description can be found in [2]. We use the term “reorganisation” to denote the process performed by SPADA.

All operations SPADA performs to come to a solution can be mapped onto graph operations that operate on an overlay network, which is called *acquaintances graph*. The acquaintances graph is defined by the agents participating in the SPP and acquaintances relationships between them, symbolised by directed links. To simplify graph operations that modify partitions, it is stipulated that each partition is a directed tree of marked links, which results in a directed forest for the partitioning as a whole.

The root of each such tree is the corresponding partition’s leader. Each partition thus has always exactly one leader. It is responsible for optimising its partition according to application-specific criteria. Each leader therefore periodically evaluates if it is beneficial to integrate new agents into its partition or to exclude members from it. The latter can be beneficial if the partition’s or an agent’s properties have changed so that the partition’s formation criteria no longer favour including the agent. Integrating and excluding agents is performed by modifying the acquaintances graph.

To decide termination, leaders periodically evaluate application-specific termination criteria formulated as predicates based on local knowledge. These predicates are constraints that can also be monitored at runtime and be used to trigger reorganisation. If these are met, the leader marks its partition as terminated. As long as a partition is terminated, its structure is not changed until a member is integrated into another partition. SPADA can thus make selective changes



(a) Flat, single layer system structure. (b) Hierarchical system structure.

Figure 1. Different system structures. The flat, single layer system structure can be created with coalition formation, clustering, or a set partitioning algorithm such as SPADA while hierarchies can be introduced when using the HiSPADA control loop in combination with one of these mechanisms.

to an existing partitioning, which is very useful in dynamic environments. It has been shown in [2] that SPADA's local decisions lead to a partitioning whose quality is within 10% of the solutions found by a centralised metaheuristic.

In many cases, it is useful that the partitions created are represented by an intermediary. In our case study, this intermediary is the AVPP. The leader of a partition instantiates a new AVPP agent after partitioning has finished. The AVPP then assumes control of all power plants in the partition.

#### IV. SELF-ORGANISING HIERARCHIES WITH HiSPADA

The original SPADA algorithm partitions the set of agents representing the entire system. It creates a flat hierarchy as shown in Fig. 1 (a). To achieve hierarchical self-organisation as depicted in Fig. 1 (b), HiSPADA uses the original SPADA algorithm to partition only a subset of agents – the *neighbourhood*. The introduction and resolution of layers is handled by the HiSPADA control loop, depicted in Fig. 2. This control loop constitutes a *partitioning control*. The HiSPADA control loop (cf. Section IV-B) runs on those intermediaries that can be reorganised, i.e., that represent partitions that can be reorganised.

##### A. Prerequisites and Interaction with SPADA

In principle, the partitioning control is independent of the concrete set partitioning algorithm used. Therefore, it would be possible to use, e.g., a particle swarm optimiser to find appropriate partitions and introduce hierarchies by using the partitioning control described here. HiSPADA's only requirements are (a) that it must be possible to limit the underlying algorithm to a certain neighbourhood and (b) that it must be possible to use application-specific formation and termination criteria to, e.g., define a minimal number of partitions to be formed. For reasons of conciseness, we will, however, focus our explanations on the control of SPADA.

The hierarchy is represented by a tree-structure formed by father-child relationships between agents. To achieve this structure, an agent is introduced that serves as the root of the tree. When the hierarchy is updated, these relationships change. SPADA has a similar but distinct notion that expresses membership in a partition. As described above, each partition formed by SPADA has a leader that is at the root of a tree of marked links. HiSPADA makes no use of these structures used by the underlying set partitioning algorithm but only of intermediaries that represent the partitions.

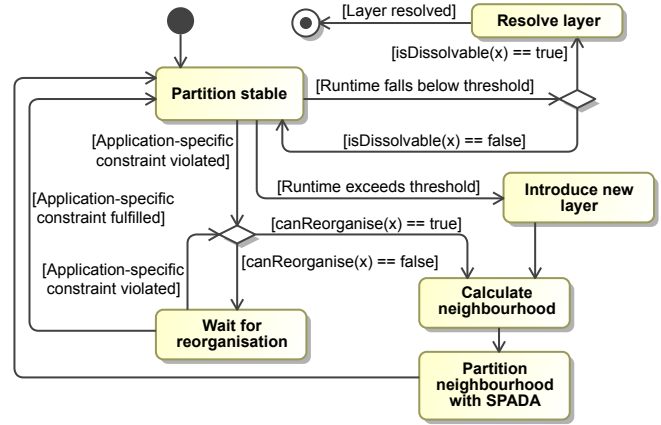


Figure 2. The HiSPADA control loop. The partitioning control running on intermediary  $x$  reacts to the violation of specific constraints, depicted here as guards, and reacts by resolving layers of the hierarchy, introducing new ones, or reorganising existing ones.

To form a hierarchy, these intermediaries are themselves partitioned. Thus, whenever an intermediary switches to another partition, the controlled agents switch with it. In the case study, a leader is always represented by an AVPP.

The HiSPADA control loop is usually dormant as long as the system is stable. It monitors the system however (more precisely: each instance of the control loop monitors the agent it runs on) and reacts to the violation of constraints. These constraints are depicted as guards on the transitions in Fig. 2. A run of SPADA and the introduction or resolution of layers is thus an attempt to restore these constraints. This behaviour conforms to the Restore Invariant Approach [15].

##### B. The HiSPADA control loop

HiSPADA has three major functional aspects: resolve an existing layer of the hierarchy by removing intermediaries; introduce a new layer in the hierarchy by creating intermediary agents; reorganise a level in the hierarchy by changing the relationships between intermediaries and the agents they control. Fig. 2 shows the control flow of HiSPADA, including these three aspects. In the following, we will use the term “hierarchy level” to denote all agents that are controlled by the same father or grandfather.

*Resolve hierarchy levels:* The resolution of existing hierarchy levels can occur for a number of application-specific reasons. In the power management example, it is triggered when the runtime of the scheduling algorithm falls below a given threshold. In that case, the AVPP that encountered this constraint violation is dissolved and all power plants are added to the father of the dissolved AVPP. In general, the predicate `isDissolvable(x)` is checked before the actual resolution. It tests whether intermediary  $x$  can be dissolved at all.

$$\begin{aligned} \text{isDissolvable}(x) &\Leftrightarrow x.\text{mayBeDissolved} \wedge \\ &x.\text{timeInExistence} \geq \text{minTimeInExistence} \end{aligned}$$

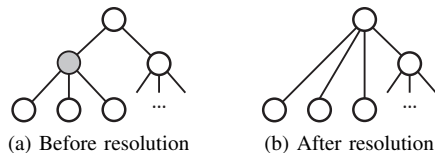


Figure 3. Resolution of a hierarchy level. The initiating agent is marked in grey. Children of the initiator become children of their previous grandfather.

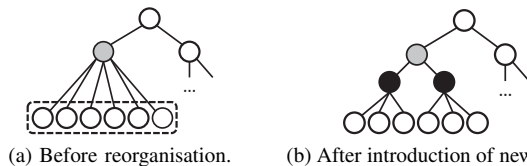


Figure 4. Introduction of a new hierarchy level. The initiating agent is marked in grey, new agents are black. The child agents of the initiating agent form the neighbourhood, marked as a dashed rectangle, that is partitioned with SPADA. New agents are introduced to form an intermediary layer.

$x.\text{mayBeDissolved}$  is false if agent  $x$  is a higher-level structure that is part of a predefined hierarchy. To avoid thrashing, it also checks if the period of grace ( $\text{minTimeInExistence}$ ) that prevents newly formed hierarchy levels to be resolved right away has already expired. The age of the agent is stored in  $x.\text{timeInExistence}$ .

Fig. 3 shows a hierarchy in which a layer is resolved. After the resolution and before the initiating agent is deleted, it informs the new father agent of the changes made. The father agent then has to react appropriately by adopting its new children and by, e.g., requesting essential data from the new children or running the control algorithm again.

*Introduce new hierarchy levels:* In the case study, new intermediaries are introduced when an AVPP requires too much time to calculate the schedule for the power plants it controls. Other applications can of course give specific conditions under which this action is performed.

When a new hierarchy level is introduced, a father agent  $f$  creates an intermediary level for its child agents. For this purpose,  $f$ 's HiSPADA control loop initialises SPADA with its child agents as the neighbourhood and a minimum number of two partitions, thus ensuring that the agents are not subsumed in just one partition. A way to guarantee this is to require that each leader knows at least one other leader. SPADA then uses this and other, application-defined criteria to create a suitable partitioning. Fig. 4 illustrates this process. The newly created agents become children of  $f$ .

*Reorganising a hierarchy level:* Whenever intermediary  $x$  introduces a new hierarchy level is introduced or detects violation of an application-specific constraints, it uses HiSPADA to reorganise a hierarchy level. For this purpose, it limits the scope of reorganisation to a certain neighbourhood, thus preventing it from crossing organisational boundaries. The original SPADA has no such limitation and reorganises the entire system.

HiSPADA has to consider some limitations when a hi-

erarchy level has to be reorganised. First of all, reorganisation can not occur while an agent's father or children are being reorganised. Otherwise, it would be possible that some agents are part of several reorganisation efforts at once, possibly resulting in changes that would violate the tree-structure of the hierarchy. This limitation is captured in the  $\text{canReorganise}(x)$  predicate that is tested before reorganisation occurs. The predicate  $\text{father}(p, q)$  denotes that  $p$  is the direct predecessor of  $q$  in the hierarchy. If an agent  $p$  currently is reorganising or is being reorganised,  $\text{reorganising}(p)$  evaluates to true.

$$\text{canReorganise}(x) \Leftrightarrow \text{isDissolvable}(x)$$

$$\wedge \neg \exists y \in \text{Agents} : \text{father}(x, y) \wedge \text{reorganising}(y)$$

$$\wedge \neg \exists z \in \text{Agents} : \text{father}(z, x) \wedge \text{reorganising}(z)$$

The second limitation is the restriction of the algorithm to neighbourhoods. In this case, the neighbourhood of the initiating agent is defined as all its children and ‘‘nephews’’, i.e., its siblings' children (cf. Fig. 5). Therefore, we define the neighbourhood  $N_x$  of an agent  $x$  as follows:

$$N_x := \{y \in \text{Agents} \mid \exists a, a_y \in \text{Agents} : \\ \text{father}(a, x) \wedge \text{father}(a, a_y) \wedge \\ \text{father}(a_y, y) \wedge \text{canReorganise}(a_y)\}$$

This neighbourhood definition ensures that only agents with fathers that can be reorganised are part of the neighbourhood, thus ensuring that predefined hierarchies or levels that are still in their period of grace are not changed. In theory, neighbourhood definitions that include more distant relatives are possible and can be implemented easily in HiSPADA. However, recursing the hierarchy up too far reduces the benefit as including more agents in the neighbourhood makes the partitioning problem more complicated and agents that are only distantly related to each other have usually been separated by HiSPADA in the course of hierarchy creation. Partitioning them closer together is not useful and might even jeopardise the system goal. Also, this approach ensures that HiSPADA tries to provide a local solution with only a limited amount of communication.

After the neighbourhood has been determined, SPADA is initialised with the set of agents in  $N_x$  and a minimal number of new partitions. Then, the partitioning algorithm is executed the same way as in the non-hierarchical case. After the algorithm has been run, existing intermediaries are reused or – depending on the number of created partitions – new ones are created or old ones removed. Fig. 5 shows an example for the reorganisation of a hierarchy level.

*Bootstrapping the system:* HiSPADA assumes very little about the initial conditions of the system. As the partitioning control runs on the agents of the system, a hierarchy will develop in the system if the constraints monitored by the control loop are violated. If there is no hierarchical structure to begin with, an initial run of SPADA

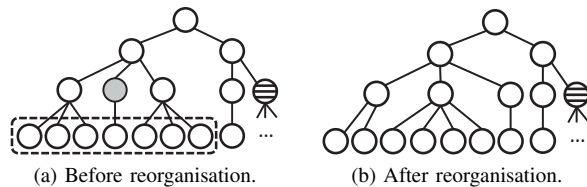


Figure 5. Reorganisation of a hierarchy level. The initiating agent (corresponding to intermediary  $x$ ) is marked in grey and the neighbourhood is marked by a dashed rectangle. The dashed agent can not be reconfigured.

can establish a partitioning on all agents in the system. This ensures that the initial partitions are suitable for the purposes of the system as SPADA uses application-specific metrics in the process. The (flat) hierarchy introduced by SPADA can easily be transformed into a tree by establishing a root agent that subsumes the intermediaries representing the newly formed partitions. This root is not dissolvable and stays at the root of the hierarchy throughout the runtime of the system. Instead of running SPADA, the system can also be partitioned randomly. In such a case, however, HiSPADA will take a while to find a suitable hierarchy if the initial partitioning did not make use of application-specific criteria.

If an organisational structure exists, corresponding partitions have to be initialised. These agents have appropriate relationships with their children to depict the organisation. HiSPADA can then work on this hierarchy by introducing intermediaries and resolving hierarchy layers formed by those intermediaries. It is therefore possible to let the partitioning control find a suitable sub-hierarchy for each of the predefined organisational entities. Of course, there is a trade-off to be made between the fine-grained depiction of existing structures and the organisational prowess of HiSPADA: if the predefined structure is too rigid, the partitioning control is not able to tackle the scalability issues it is intended for.

## V. EVALUATION

HiSPADA has been tested in a simulation environment for power management applications, simulating 435 power plants and 12 consumers. In the application, a constraint optimisation problem is solved to assign the overall power demand to individual power plants while minimising the gap between the power demand and the scheduled power production. The scheduling problem is NP-complete and the runtime of the solution algorithm increases polynomially with the number of agents involved. Power plant models and power demand are based on real-world data. Such an application is a typical operational scenario for HiSPADA, since the main task is computationally much more expensive than hierarchy formation and can be hierarchically decomposed.

If no hierarchies exist (Scenario A), schedules are created by a centralised control system for all 435 power plants. If the original SPADA is used to establish a flat hierarchy in the system (Scenario B), the power plants' schedules are created by the respective AVPPs. The schedules of the AVPPs are

TABLE I  
EVALUATION RESULTS FOR SCALABILITY AND SYSTEM STABILITY

	Sc. A	Sc. B	Sc. C	Sc. D
Max. sequential runtime of scheduling in ms	3624	925	499	484
	$\pm 55$	$\pm 638$	$\pm 220$	$\pm 213$
Avg. height of hierarchy	-	2	4.99	3.52

again created by a root AVPP, as described in Section IV. In case HiSPADA can work without predefined organisations (Scenario C) and within a predefined hierarchy with an initial height of 3 (Scenario D), schedules are created in each AVPP on different levels of the hierarchy. We define the height of the hierarchy as the length of the longest downward path to a leaf from the root.

The evaluation's focus is on performance indicators for scalable operation. The most important criterion is the average runtime of the scheduling algorithm: if the partitioning control works, the overall runtime should be reduced when hierarchies are introduced while maintaining stable system operation. The latter can be measured by comparing the gap between power production and demand as calculated by a central authority with the sum of the gaps calculated on the different levels of the hierarchy. As the deviation between these numbers was minimal and the quality of the scheduling thus was not impaired, we did not include it in Table I which lists the results for runtime and height of the hierarchy.

All results were averaged over 100 simulation runs for each scenario. The maximum sequential runtime provided in the table is the maximum of the sums of the runtime in each branch originating from the root. After the root node has calculated a schedule for its direct children, these in turn schedule their children. The scheduling is recursively performed on lower levels until the schedules for the physical power plants are created. As each branch performs the scheduling concurrently, no overhead is introduced.

The evaluation results in Table I show that the average maximum sequential scheduling time is reduced significantly with the introduction of hierarchies. In Scenario B, the high standard deviation can be explained by the variation of partition sizes created by SPADA in different runs. HiSPADA ensures more homogeneous partitions and thus less variation in the scheduling times. At the same time, the average height of the hierarchy has a direct effect on the interaction between SPADA and the partitioning control. The deeper the hierarchy, the smaller the neighbourhoods on which SPADA is executed but the more scheduling runs are performed, making it necessary to find a proper balance. Since SPADA scales very well for small sets (see Fig. 6) and HiSPADA reduces set sizes, hierarchies additionally reduce the overhead for restructuring the system. Very large systems will profit most from these reductions. The results for Scenario D also show that HiSPADA can work effectively with predefined organisations. If necessary, HiSPADA adds additional layers to further decompose the computation task and thus reduces scheduling runtime.

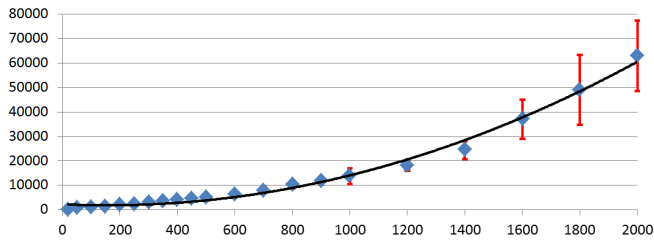


Figure 6. Runtime of SPADA for different neighbourhood sizes. The algorithm scales nearly linearly for sets below 1000 agents. Hierarchies limit the neighbourhood size and allow to leverage this property.

## VI. DISCUSSION AND FUTURE WORK

We have introduced HiSPADA, a hierarchical partitioning control that – in combination with a set partitioning algorithm such as SPADA – forms hierarchical layers in large multi-agent systems. The formation of hierarchies is driven by scalability metrics provided by the application. We gave an example limiting the duration of scheduling in a power management scenario. HiSPADA introduces layers to limit the number of agents controlled by an intermediary and removes layers if they are no longer necessary. As it only removes layers that have been introduced by the partitioning control itself, it respects representations of existing organisational structures. The evaluation shows that scalability issues can be tackled this way and that the benefits of the hierarchy formation outweigh the overhead of the partitioning control.

In systems in which a hierarchical task decomposition is not possible, HiSPADA can be useful to establish a hierarchy in which control can be delegated to sub-ordinate levels or to subsume certain responsibilities on a higher level of the hierarchy. In such cases, normative systems are helpful to regulate the delegation of power and responsibilities at the different levels [16], a topic considered for future work.

Finally, an important issue in heterogeneous multi-agent systems that use hierarchical task decomposition are the models used in the decision making process. An hierarchical scheduling mechanism like the one used in the example requires models of the power plants that are scheduled. Since power management systems are long-lived and consist of a variety of different power plants of different types and from different vendors, the models used in the process are only known at runtime. Therefore, model synthesis is necessary to form a new, combined model from the individual parts on each hierarchical layer whenever the structure of that layer changes. Likewise, as each layer in the system is in turn regarded as a power plant, the combined model has to be abstracted so it can be used on a higher layer to make control decisions. Both the processes of model synthesis and abstraction are important topics of future work.

### ACKNOWLEDGMENT

This research is partly sponsored by the German Research Foundation (DFG) in the project “OC-Trust” (FOR 1085).

## REFERENCES

- [1] B. Horling and V. Lesser, “A survey of multi-agent organizational paradigms,” *The Knowledge Engineering Review*, vol. 19, no. 04, pp. 281–316, 2004.
- [2] G. Anders, F. Siefert, J.-P. Steghöfer, and W. Reif, “A decentralized multi-agent algorithm for the set partitioning problem,” in *PRIMA 2012: Principles and Practice of Multi-Agent Systems*, ser. Lecture Notes in Computer Science, I. Rahwan, W. Wobcke, S. Sen, and T. Sugawara, Eds. Springer Berlin / Heidelberg, 2012, vol. 7455, pp. 107–121.
- [3] G. Anders, F. Siefert, J.-P. Steghöfer, H. Seebach, F. Nafz, and W. Reif, “Structuring and Controlling Distributed Power Sources by Autonomous Virtual Power Plants,” in *Proc. of the Power & Energy Student Summit 2010*, October 2010, pp. 40–42.
- [4] A. Koestler, *The ghost in the machine*. London: Hutchinson, 1967.
- [5] A. Colombo, R. Schoop, and R. Neubert, “An agent-based intelligent control platform for industrial holonic manufacturing systems,” *Industrial Electronics, IEEE Transactions on*, vol. 53, no. 1, Feb. 2005, pp. 322–337.
- [6] H. Prothmann, S. Tomforde, J. Branke, J. Hähner, C. Müller-Schloer, and H. Schmeck, “Organic traffic control,” in *Organic Computing – A Paradigm Shift for Complex Systems*, ser. Autonomic Systems, C. Müller-Schloer, H. Schmeck, and T. Ungerer, Eds. Springer Basel, 2011, vol. 1, pp. 431–446.
- [7] S. Bandyopadhyay and E. Coyle, “An energy efficient hierarchical clustering algorithm for wireless sensor networks,” in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, 2003, pp. 1713–1723.
- [8] V. Kotov, “Systems of systems as communicating structures,” in *Object-oriented technology and computing systems re-engineering*, H. Zedan and A. Cau, Eds. Chichester, USA: Horwood Publishing, Ltd., 1999, pp. 141–154.
- [9] W. H. Manthorpe, “The emerging joint system of systems: A systems engineering challenge and opportunity for APL,” *Johns Hopkins APL Technical Digest*, vol. 17, no. 3, 1996, p. 305.
- [10] A. P. Sage and C. D. Cuppan, “On the systems engineering and management of systems of systems and federations of systems,” *Information, Knowledge, Systems Management*, vol. 2, no. 4, 2001, pp. 325–345.
- [11] T. Lenaerts, D. Chu, and R. Watson, “Dynamical hierarchies,” *Artificial Life*, vol. 11, no. 4, 2005, pp. 403–405.
- [12] A. Dorin and J. McCormack, “Self-assembling dynamical hierarchies,” in *Proceedings of the 8th International Conference on Artificial life*, ser. ICAL 2003. Cambridge, MA, USA: MIT Press, 2003, pp. 423–428.
- [13] V. Valev, “Set partition principles revisited,” in *Advances in Pattern Recognition*, ser. LNCS. Springer, 1998, vol. 1451, pp. 875–881.
- [14] T. Rahwan, S. D. Ramchurn, N. R. Jennings, and A. Giovannucci, “An Anytime Algorithm for Optimal Coalition Structure Generation,” *Journal of Artificial Intelligence Research*, vol. 34, 2009, pp. 521–567.
- [15] F. Nafz, H. Seebach, J.-P. Steghöfer, G. Anders, and W. Reif, “Constraining Self-organisation Through Corridors of Correct Behaviour: The Restore Invariant Approach,” in *Organic Computing – A Paradigm Shift for Complex Systems*, ser. Autonomic Systems. Springer Basel, 2011, vol. 1, pp. 79–93.
- [16] A. Artikis, M. Sergot, and J. Pitt, “Specifying norm-governed computational societies,” *ACM Transactions on Computational Logic (TOCL)*, vol. 10, no. 1, 2009, pp. 1–42.

## Methodology of Training and Support for Urban Search and Rescue With Robots

Janusz Bedkowski, Karol Majek, Igor Ostrowski,  
Paweł Musialik, Andrzej Masłowski  
Institute of Mathematical Machines IMM  
Warsaw, Poland  
e-mail: januszbedkowski@gmail.com,  
karolmajek@gmail.com, iostrowski@wp.pl,  
pawelmus@op.pl, a.maslowski@imm.org.pl

Antonio Coelho  
FEUP  
INESC TEC  
Porto, Portugal  
e-mail: acoelho@fe.up.pt

Artur Adamek  
Department of Geodesy and Cartography  
Warsaw University of Technology  
Warsaw, Poland  
e-mail: a.adamek@gik.pw.edu.pl

Geert De Cubber  
Department of Mechanics MECA  
Royal Military Academy  
Brussels, Belgium  
e-mail: geert.de.cubber@rma.ac.be

**Abstract**— A primordial task of the fire-fighting and rescue services in the event of a large crisis is the search for human survivors on the incident site. This task, being complex and dangerous, often leads to loss of lives. Unmanned search and rescue devices can provide a valuable tool for saving human lives and speeding up the search and rescue operations. Urban Search and Rescue (USAR) community agrees with the fact that the operator skill is the main factor for successfully using unmanned robotic platforms. The key training concept is “train as you fight” mentality. Intervention troops focalize on “real training”, as a crisis is difficult to simulate. For this reason, in this paper a methodology of training and support for USAR with unmanned vehicles is proposed. The methodology integrates the Qualitative Spatio-Temporal Representation and Reasoning (QSTRR) framework with USAR tools to decrease the cognitive load on human operators working with sophisticated robotic platforms. Tools for simplifying and improving virtual training environment generation from life data are shown.

**Keywords** – USAR; robot; training and support; qualitative reasoning

### I. INTRODUCTION AND RELATED WORK

The major factor of training determining the acceptance of the USAR community is for the learning process of different tools to conclude within one week, which is the duration of a typical Search and Rescue (SAR) training course. For this reason, it can be difficult to adopt a new training technology if the training requires more than this duration. Training and support is related with effective tasks the robots would be able to do.

In this context, CRASAR, the Centre for Robot-Assisted Search and Rescue from the Texas A&M University, compiled a list of distinct activities for rescue robots [1][2].

The activities and tasks relevant to USAR with unmanned vehicles are: Area reduction, in the early stages of a disaster, there is generally a large area which must be labeled as “possibly affected”. To deploy the USAR teams correctly it is required to assess as soon as possible which areas are more affected and which areas are less affected. Sectorization, the disaster area must be subdivided into sectors. Search, robotic tools could speed up the search for victims. Reconnaissance and mapping, robotic tools can help to increase the common operation picture and situational awareness of the deployed teams by mapping the terrain. Rubble removal, robots can remove heavy rubble faster than humans. Debris estimation, after a major crisis, there is in general a lot of debris lying around, impeding the proper deployment of rescue operators and goods. Robots could help with a quicker and better assessment of the most affected zones. Structural inspection and shoring, USAR teams need to assess the structural integrity of a building and may help to stabilize it before entering. On site medical assessment and intervention, robots can provide medical personnel means to inspect a victim remotely via video or audio and to provide the victim life support. Evacuation of casualties, robots may act as carriers to evacuate victims from the disaster area. Acting as mobile beacon or repeater, Robots can help with extending the mobile communication range, by acting as a repeater. Over the horizon applications, Often, SAR teams want to know the damage in a nearby village/suburb/town. It would be highly convenient to be able to send a light Unmanned Aerial System (UAS).

Within the context of mentioned tasks relevant to USAR with unmanned vehicles it is observed that most of these activities can be modeled using a QSTRR [3]. QSTRR proposes the interaction with the environment through a spatial design task. The problem of providing intelligent spatial decision-making capabilities is related to the

framework of Multi-Modal Data Access for Spatial Assistance Systems [4]. This framework shows a key concept of spatial assistance systems by focusing on multi-perspective semantics, qualitative and artefactual abstractions, industrial conformance and interoperability. Authors also provide examples of use for distinct application domains, which was an important input for the developed methodology of training and support. A core concept of artefactual abstractions is to provide qualitative categorization into functional, operational and range spaces [5]. Functional space is a region of space within which a person must be located in, in order to interact with the object or to employ the object for its intended function. Operational space is a region of space that an object requires to perform its intended function. Range space is a region of space where the object operates as a result of performing its intended function. These spaces can model robotic activities in disaster zones.

Another important aspect used in the proposed methodology of training and support is related to georeferencing of terrestrial laser-scanner data for applications in architectural modeling [6]. Using a 3D laser we can obtain accurate 3D models of the disaster zone, which is the key concept in the proposed USAR-training and support application. From the USAR community point of view, the use of laser scanning can provide frequently updated, accurate and reliable data. In [7] Kurt3D, data from KINECT sensor and Riegl VZ-400 3D scanner are shown, which gives an impression that the State of the Art (SoA) offers efficient mobile platforms equipped with advanced sensors for obtaining accurate maps, that could be used in USAR. Some of these maps are used for semantic objects identification [8]. Using semantic information extracted from 3D laser data is a relatively recent research topic in mobile robotics. In [8] a semantic map for a mobile robot was described as a map that contains, in addition to spatial information about the environment, assignments of mapped features to entities of known classes. In [9] a model of an indoor scene is implemented as a semantic net. This approach is used in [10] where a robot extracts the semantic information from 3D models built from a laser scanner.

The proposed methodology of training and support is adopting procedural modeling tools [11][12] that can be used for the generation of virtual urban environments, reducing both the amount of interaction needed as well as modeling effort. These techniques can be enhanced to generate accurate models that incorporate the semantic data existing in Geographic Information System (GIS) [13][14]. On the other hand, municipalities often store several semantic data regarding the urban features populating the territory in a GIS. Procedural modeling of urban environments originates from the use of L-Systems [15]. The limitations of this mathematical tool led to the development of the CGA Shape [16], a shape grammar capable of producing extensive architectural models with high detail. The implementation of the CGA Shape is integrated in the CityEngine framework. The same limitation led to the development of Geospatial L-Systems [17], an extension of parametric L-Systems which incorporates spatial awareness. This approach combines the

ability of data amplification provided by the L-Systems with the geospatial awareness of GIS. Approach proposed in this paper combines procedural modeling tools, Qualitative Spatio-Temporal Representation and Reasoning framework and modern laser scanning to provide methodology for training and support for robotic USAR application.

The paper is organized as follows: Section II describes the use of QSTRR framework for purpose of semantic modeling, Section III concentrates on procedural modeling of urban environments, Section IV, V and VI describe data processing and conceptualization, experiments conducted are described in Section VII, finally Section VIII contains conclusion and future work description.

## II. SEMANTIC MODEL OF USAR OPERATION

To model an USAR operation, the QSTRR is proposed. The main element is an ontology. As a representation vocabulary it is specialized to the domain of physical/functional entities in real structured environment. It allows to build a model of an environment using qualitative spatio-temporal or quantitative (depends on the need) representation. An ontology (O) is composed of several entities: a set of concepts (C), a set of relations (R), a set of axioms (A) (e.g., transitivity, reflexivity, symmetry of relations), a concepts' hierarchy (CH), a relations' hierarchy (RH), a set of spatio-temporal events (Est), what can be formulated as following definition:

$$O = \langle C; R; A; CH; RH; Est \rangle \quad (1)$$

A concept is defined as a primitive spatial entity described by a shape (S) composed of polygons in 3D space associated with a semantic label (SL). Ontology distinguishes two different types of attributes that can be assigned to a concept, quantitative (Aqn) and qualitative (Aql). Four values of qualitative attribute (entity function) are listed: real physical object, functional space, operational space, range space. Functional, operational and range spaces are related with spatial artifacts that describe the USAR environment and robotic devices such as sensors and actuators. Quantitative attributes are related with physical properties of spatial entities and are as follows: location, mass, center of mass, moment of inertia (how much resistance there is to change the orientation about an axis), material (friction, restitution). Therefore, the definition of the concept (C) is formulated as:

$$C = \langle S; Aqn; Aql; SL \rangle \quad (2)$$

The set of relations (R) is composed of quantitative and qualitative spatial relations. For topological spatial relations



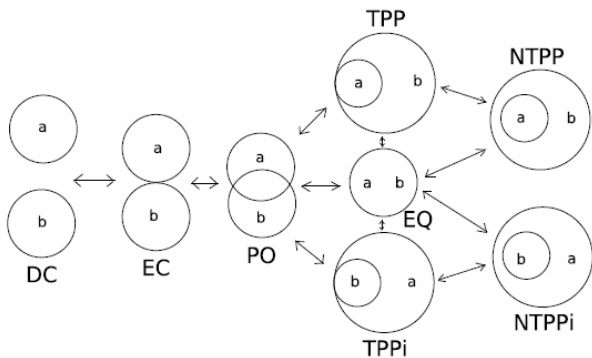


Figure 1. The relations of RCC-8 calculus (conceptual neighborhood).

(qualitative) the Region Connected Calculus (RCC) is proposed. RCC is a formalism for spatial reasoning that takes regions of space (shapes) instead of points of classical geometry as primitives. One particular prominent reasoning system is a system of topological relations called RCC8 (the relations of RCC-8 calculus and conceptual neighborhood is shown in Figure 1, therefore the ontology includes eight different topological relations between two regions (in our case shapes): disconnected (DC), externally connected (EC), partial overlap (PO), equal (EQ), tangential proper part (TPP) and its inverse (TPPi), non-tangential proper part (NTPP) and its inverse (NTPPi).

Quantitative spatial relations are a way to constrain the way entities move relative to each other. The ontology defines the following constraints: origins locked, orientations locked; origins locked, orientations free; free rotation around one axis; sliding. Ontology provides a mechanism for building world models that assume spatio-temporal relations in different time intervals (in other words: world models that can capture changes) for the representation of spatio-temporal knowledge used for spatio-temporal reasoning. Chosen temporal representation takes temporal intervals as a primitive, therefore ontology defines qualitative spatio-temporal events (Est) related with topological spatial relations RCC-8:

onEnter (DC->EC->PO),  
 onLeave (PO->EC->DC),  
 onStartInside (PO->TPP->NTPP),  
 onStopInside (NTPP->TPP->PO).

These four qualitative spatio-temporal events can be used to express most important spatio-temporal relations that can be held between two concepts in different intervals of time. To store the instances of ontology-based elements (defined on the conceptual level) an instance base ( $IB^O$ ) is defined:

$$IB^O = \langle I_C^O ; I_R^O ; I_{Est}^O \rangle \quad (3)$$

where:  $I_C^O$  contains instances of concepts C,  $I_R^O$  contains instances of relations R,  $I_{Est}^O$  contains instances of spatio-temporal events. Semantic model is defined as a pair:

$$SM = \langle O ; IB^O \rangle \quad (4)$$

where: O is an ontology and  $IB^O$  is an instance base related to ontology O. Ontology is known a-priori but instance base is being updated during USAR operation. Semantic model is



Figure 2. City model generated from municipal GIS

a core concept for support system. The projection of the semantic model onto 3D space is defined as 3D semantic map, and the projection of the semantic model onto 2D space is defined as 2D semantic map. Semantic maps are useful visualization tools for increasing the awareness of search teams concerning the global operational picture of the USAR scenario.

### III. PROCEDURAL MODELING TOOLS FOR THE GENERATION OF VIRTUAL URBAN ENVIRONMENTS

Procedural methods require that the user, capturing the knowledge about the modeling process, introduces some guidelines and rules. This is time consuming and can be difficult to develop in a crisis situation. By taking advantage of the semantic model for USAR operation this approach uses generic production rules to amplify available data from existing shapes or general information.

In this sense, some methods have already been conceived, but as far as control is concerned, procedural ways still lack powerful picking and manipulation facilities to apply geometric operations. This motivates the development of more advanced methodologies for such control.

Based on PGCAD API [12], a solution for geometric manipulation in procedural modeling tools that incorporates spatial awareness and semantic control, it is possible to achieve more powerful control over geometric entities based on their properties and sequential application of modeling operations, therefore allowing a greater, faster and more intuitive approach for geometry generation. This is achieved through its intuitive topological structure, which features a set of properties, such as scope, spatial awareness and semantic information. The modeling processes can be massively applied to sets of shapes, yet act according to each individual shape's properties. This allows a more customized control, as well as successive tracking, which induce a greater, faster and more intuitive approach for geometry generation.

IV. REAL TIME 3D DATA PROCESSING

The goal of support system is to increase the awareness of rescue team concerning a disaster sector. For this reason, a data filtering and registration tool is developed [18]. Figures 3 and 4 demonstrate the computation of 3D clouds obtained with a modern 3D laser scanner (each cloud contains 1,5 million of data points).

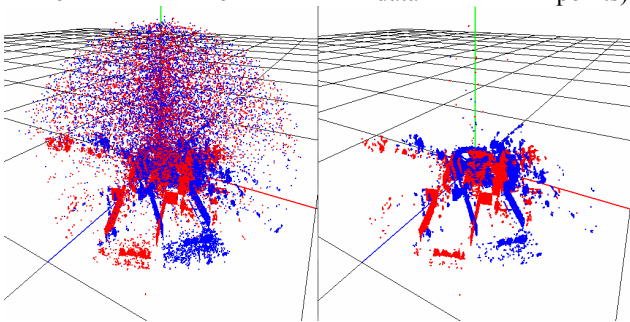


Figure 3. The filtering of 2D cloud of points.

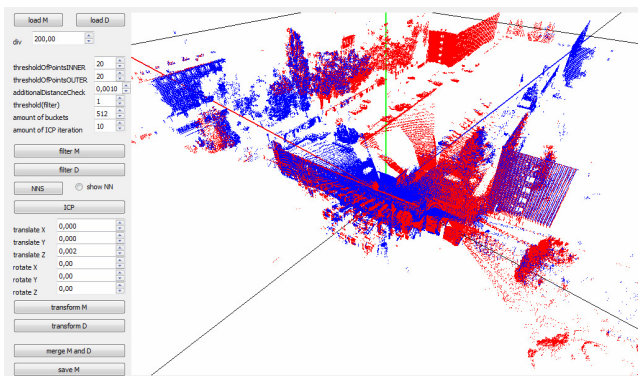


Figure 4. The registration of 3D cloud of points.

The computation time of filtering is on average 1,2 second, the registration takes in average 2 minutes assuming computational help of the ICP (Iterative Closest Point) algorithm. It is important to emphasize that the proposed approach can register up to 65million of points within 30 minutes.

V. MANUAL CONCEPTUALISATION

The proposed support system provides the functionality of manual conceptualization based on the registered 3D cloud of points (Figure 5). Rescue team members will have the possibility to assign semantic labels to the observed objects.

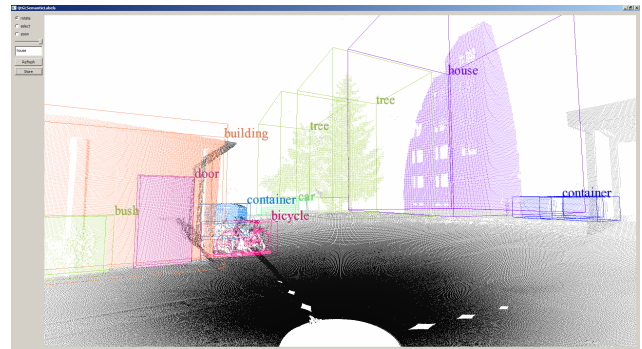


Figure 5. The view of HMI for manual conceptualisation.

These labels correspond to the spatial regions in the 3D scene will generate additional semantic concept in the semantic model of global operational picture that is distributed over USAR system.

VI. AUTOMATIC CONCEPTUALISATION

To decrease the cognitive load on rescue team members, the proposed support and training system provides a mechanism for automatic conceptualisation based on artificial intelligence techniques. In its current form, the system classifies each point into one of four classes: ground, building, vegetation, unclassified. The assignment is made by analysing normal vectors of points. The first step concentrates on finding the post populous horizontal plane, below the origin point of the scan. To achieve this, a RANdom SAmple Consensus (RANSAC) [19] method is used. After that, each point whose distance to the found plain is lower then R is classified as ground. Those points are then filtered out from the data 3D point cloud. Remaining points are grouped into cells of 2x2 meters, in the horizontal plane. They main assumption, on which classification between vegetation and buildings is made is that buildings are mostly regular plains, whereas trees and bushes are mostly irregular group of points. Therefore, a cell that holds vegetation points should have a roughly uniform distribution of directions of normal vectors, and cell that holds building walls should have most normal vectors pointing in one direction. Based on that reasoning the cells are classified in two groups: potential building or potential vegetation. The cell is considered potential building if normal vectors of at least half of the points in the cell point in roughly the same direction. In such cell RANSAC algorithm is used to see if such a plane can be found that at least half of the points in the cell is in N-distance of it. If this condition is met, the cell is classified as a building. Every cell that does not meet this condition is added to the group of potential vegetation. The vegetation hypothesis is checked by counting the dominant direction of normal vectors of each point in the cell. If the distribution of normal vectors directions is roughly uniform, the cell is considered vegetation. The cells that do not meet this condition are classified as unclassified. After initial classification, every cell is checked, by comparing it with it's neighbors.

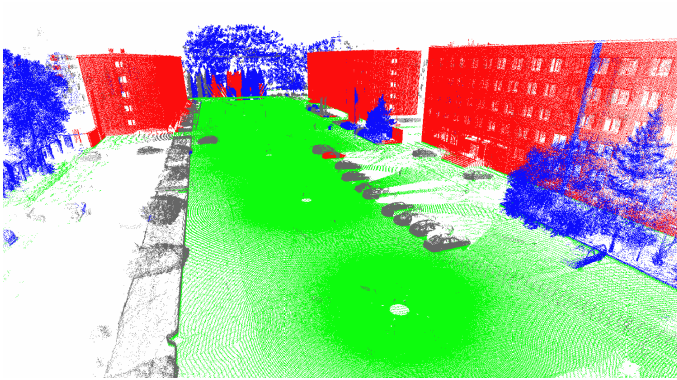


Figure 6. The view of HMI for the visualisation of automatic conceptualisation.

The results of the algorithm are shown in Figure 6. Classes of points are shown by color: green-ground, red-building, blue-vegetation, gray-unclassified. Computation time for 15 million points cloud for this particular experiment is 50s.

VII. EXPERIMENT

To proof the concept of qualitative reasoning used in USAR with robots, we validate the methodology in the environment shown in Figure 7. Each cell corresponds to a 10x10 meter rectangle. Therefore, the global operational picture covers region of 200x200 meters. The semantic model contains 3D shapes of the scanned real objects, robot goals (starting point, middle goal, and destination goal), robot path – rectangular prisms – 3D shapes between goals, robot shape and artificial cameras (Figure 8). The projections of the semantic model onto 3D and 2D spaces are shown in Figures 9 and 10. These projections are defined as semantic maps, which are to be visualized for rescue team members to increase their awareness of global operational picture. The qualitative reasoning capabilities for the given example can be demonstrated using the following request-respond pairs.

**Request:** where is the robot?

**Qualitative respond:** robot is inside the path.

**Quantitative respond:** robot's GPS(Global Positioning System) coordinates.

When human operator will try to omit the path support system will respond:

**Qualitative respond:** robot is going out of the path.

**Quantitative respond:** robot's GPS coordinates.

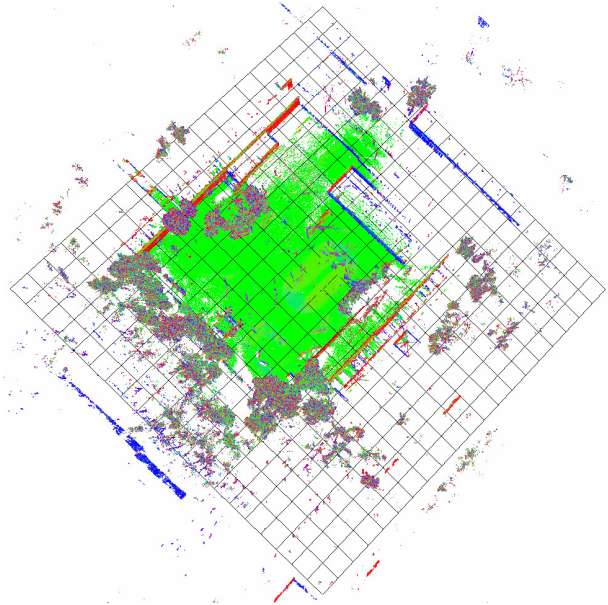


Figure 7. Testing environment - the registreted 3D cloud of points.

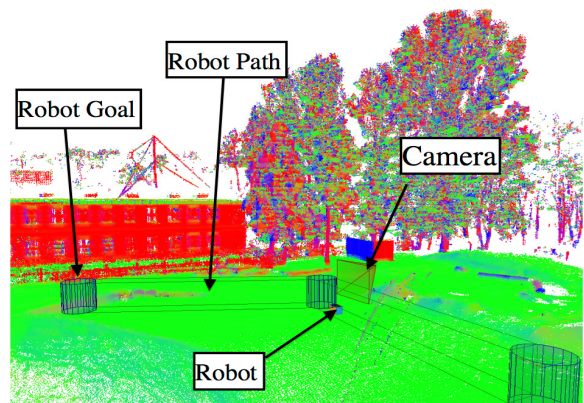


Figure 8. Visualisation of the semantic model.

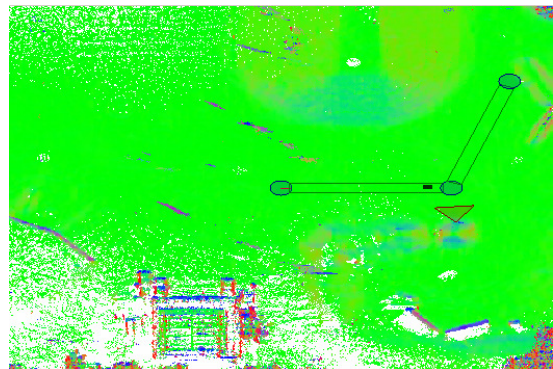


Figure 9. 2D semantic map of semantic model from Figure 7.

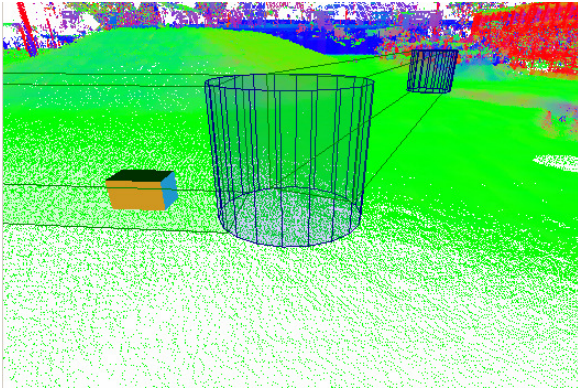


Figure 10. 3D semantic map of semantic model from Figure 7. The view point - range space of artificial camera.

Based on these simple examples it can be observed that qualitative reasoning capabilities are very useful for rescue team members because of its simple form. It is important to simplify the use of advanced technologies which can help to introduce new modules into the USAR community.

### VIII. CONCLUSION

In this paper, a methodology for training and support for Urban Search and Rescue with robots is shown. The methodology integrates the QSTRR framework with USAR tools for decreasing the cognitive load on human operators working with sophisticated robotic platforms. The goal was to develop software component for filtering and registering 3D data acquired with modern 3D laser scanner (providing millions of points in a single 3D scan). Registered data is manually or automatically conceptualized, providing a semantic model associated with a global operational picture. Through the modification of this semantic map, we can model complex USAR scenarios, assuming the usage of the robotic platforms.

### ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°285417.

### REFERENCES

- [1] J. Burke, R. Murphy, E. Rogers, V. Lumelsky, and J. Scholtz, "Final report for the RARPA/NSF interdisciplinary study on Human-Robot Interaction" IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews, Vol.34, No. 2, 2004, pp. 103-112.
- [2] J. Carlson, R. Murphy, and A. Nelson, "Follow-Up analysis of mobile robot failures," CRASAR Technical Report - TR2004-10, 2004.
- [3] J. Będkowski, "Intelligent mobile assistant for spatial design support," Journal of Automation in Construction, 2012, doi: <http://dx.doi.org/10.1016/j.autcon.2012.09.009>
- [4] C. Schultz and M. Bhatt, "A multi-modal data access framework for spatial assistance systems: use-cases with the building information model (bim/ifc)," ISA, Proc. 2nd ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness, 2010, pp. 39-46.
- [5] M. Bhatt, F. Dylla, and J. Hois, "Spatio-Terminological interference for the design of ambient environments," in Conference on Spatial Information Theory(COSIT'09), K. S. Hornsby, C. Claramunt, M. Denis, and G. Ligozat, pp. 371-391, Springer-Verlagm, 2009.
- [6] S. Schuhmacher and J. Bohm, "Georeferencing of terrestrial laserscanner data for applications in architectural modeling," 3D-ARCH 2005: Virtual reconstruction and visualization of complex architectures, Mestre-Venice, Italy, 22-24 August, 2005, Univesitat Stuttgart, 2005, URL: <http://elib.uni-stuttgart.de/opus/volltexte/2007/3256>
- [7] J. Elseberg, D. Borrmann, and A. Nuchter, "Efficient processing of large 3d point clouds," Proc. XXIII International Symposium on Information, Communication and Automation Technologies(ICAT11), Srajevo, Bosnia, 2011, pp. 1-7
- [8] A. Nuchter and J. Hertzberg, "Towards semantic maps for mobile robots," Robot. Auton. Syst., vol. 56, no. 11, 2008, pp. 915-926, doi: 10.1016/j.robot.2008.08.001. URL: <http://dl.acm.org/citation.cfm?id=1453261.1453481>
- [9] O. Grau, "A scene analysis system for the generation of 3-d models," NRC '97, Proc. International Conference on Recent Advances in 3-D Digital Imaging and Modeling, IEEE Computer Society, Washington, DC, USA, 1997, p. 221-228.
- [10] A. Nuchter, H. Surmann, K. Lingemann, and J. Hertzberg, "Semantic scene analysis of scanned 3d indoor environments," Proc. Eighth International Fall Workshop on Vision, Modeling, and visualization (VMV 03), 2003, pp. 665-673.
- [11] P. B. Silva, A. Coelho, R. Rodrigues, and A. A. de Sousa, "A procedural geometry modeling API," Proc. GRAPP & IVAPP 2012, Italy, SciTePress 2012, 24-26 February, 2012, pp. 129-134.
- [12] P. B. Silva and A. Coelho, "Procedural modeling for realistic virtual worlds development," Journal of Virtual Worlds Research, vol. 4, no. 1, 2011, doi:10.4101/jvwr.v4i1.2109
- [13] D. Jesus, A. Coelho, C. Rebelo, A. Cardoso, and A. A. Sousa, "A pipeline for procedural modelling from geospatial data," Proc. Eurographics 2012, pp. 9-10, 2012.
- [14] T. Martins, P. B. Silva, A. Coelho, and A. A. Sousa, "An urban ontology to generate collaborative virtual environments for municipal planning and management," Proc. GRAPP 2012 -7th International Conference in Computer Graphics Theory and Applications, pp. 507-510, 2012.
- [15] Y. I. H. Parish and P. Müller, "Procedural modeling of cities." Proc. 28th annual conference on Computer graphics and interactive techniques(SIGGRAPH '01), ACM, New York, USA, 2001, pp. 301-308.
- [16] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. van Gool, "Procedural modeling of buildings" ACM Trans Graph., vol. 25, July 2006, pp. 614-623.
- [17] A. Coelho, M. Bessa, A. A. Sousa, and F. N. Ferreira, "Expeditious modeling of virtual urban environments with geospatial l-systems," Computer Graphics Forum, vol. 26, no. 4, 2007, pp. 762-782.
- [18] J. Będkowski, A. Masłowski, and G. De Cubber, "Real time 3D localization and mapping for USAR robotic application," Industrial Robot: An International Journal, vol. 39, no. 5, 2012, pp. 464-474.
- [19] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Comm. of the ACM, vol. 24, no. 6, june 1981, pp. 381-395, doi :10.1145/358669.358692 .

## Development of Ontology Based Framework for Information Security Standards

Partha Saha  
MIS Group  
IIM Calcutta  
Kolkata, India  
shree.partha.saha@gmail.com

Ambuj Mahanti  
MIS Group  
IIM Calcutta  
Kolkata, India  
am@iimcal.ac.in

B.B. Chakraborty  
Finance and Control Group  
IIM Calcutta  
Kolkata, India  
bbc@iimcal.ac.in

Avinash Navlani  
MIS Group  
IIM Calcutta  
Kolkata, India  
avinashnvl8@gmail.com

**Abstract**— E-Business Management and associated risk mitigation of organizational resources have become a major challenge for the organizations in light of increasingly global and integrated digital economies. Our research focuses on information security in e-Business management. We consider, in particular, the domain of banking. The banking sector, being highly regulated, poses plethora of challenges in terms of compliance of organizational practices with regulatory standards such as Basel III, CobiT 4.1 and ISO17799. An automated compliance auditing solution to the existing manual auditing is highly desirable from management’s standpoint due to considerable savings in cost and time. In this paper, we envisage a new paradigm where ontology based information model is used in an automated compliance auditing application. It performs compliance checking to verify if actual banking practices are following information security standards and whether discrepancies between security standards and actual banking practices call for qualified, adverse, disclaimer or piecemeal opinion by the information security auditor, while investigating efficacy of information security standards employed in banking domain.

**Keywords**-Information Security; Compliance Auditing; Risk Management; Indian Banking Regulation .

### I. INTRODUCTION

Compliance Management (CM) is a business process that concerns organizations of different magnitude and size. It deals with the process of checking the organization practices with the regulatory compliance policies and business guidelines in an integrated and networked environment. This process is a continuous and labour intensive task that involves business and management’s commitments, time and resources in demonstrating organizational alignment, adherence and compliance to the prevailing regulations and best practices (such as Sarbanes Oxley [9] , HIPPA [12], Basel III [21], CobiT4.1 [20] ). The importance of compliance is underscored from renowned corporate frauds like Enron and WorldCom [23]. In this paper, we are analyzing the application of ontology to (semi) automate compliance auditing process (a process which is till now completely manual). We try to address the fundamental research question of how we may segregate the compliance rules and regulations of a standard (CobiT4.1) and organization practice (that of a bank X in India, which is implementing CobiT4.1) into two different ontology layers (source and target ontology, respectively) which may facilitate automated compliance auditing. By comparing two ontologies, performance evaluation of the organization vis-

à-vis source regulations may be ascertained. Although here we are applying our methodologies upon specific case of application of CobiT4.1 in banking domain, the application of the methodology is independent of any specific domain or standard.

The paper is organized as follows. Section II briefly discusses GRC (Corporate Governance, Risk Management and Compliance Auditing) while Section III undertakes literature survey. Section IV briefly expounds methodological framework of ontology based compliance auditing, while Section V discusses the compliance measurement in information security standards. Section VI applies the framework for an Indian bank X, while Section VII calculates the compliance metric for the bank X and discusses various scenarios. The paper concludes with Section VIII.

### II. CORPORATE GOVERNANCE, RISK MANAGEMENT AND COMPLIANCE AUDITING (grc)

In this section, we will be briefly discussing about corporate governance, risk management and compliance auditing related to banking sector.

#### A. IT Governance and Regulatory Standards

IT governance is a part of overall corporate governance process of any organization. Aligning Information Technology with the strategic goals of the organization, delivering promised value to the stakeholders, optimum utilization of critical IT resources, undertaking risk management and strict performance monitoring are some of the cornerstone of IT governance. In this subsection, we present two well-known regulations and standards.

##### 1) BASEL III

Basel III, a brainchild of BCBS (Basel Committee of Banking Supervision) [21], (which came into effect starting January 1st, 2013) is equipped with twin legal instruments namely: Directives and Regulations. The Directive contains four cardinal principals, namely: (i) Increased Governance (ii) Capital Buffer (iii) Increased Supervision and (iv) Sanctions. The Regulation part contains five important sections, namely: (i) Definition of Capital (ii) Credit Risk from Counterparty (iii) Risk of Liquidity (iv) Single Rule Book and (v) Leverage Ratio [21]. With these set of stipulations, BCBS is trying to strengthen corporate governance, enhance banks’ risk management capability, transparency and disclosure [21].

## 2) Control Objectives for Information and related Technology (CobiT)

CobiT has been developed by the IT Governance Institute and it provides reference framework for good IT standards and practices. It is a tool that is used by the Information Security auditors and practitioners alike. CobiT includes a framework that responds to the management's need for adequate control and measurement of IT by providing tools to standardize, assess and measure the organization's IT resource and capabilities vis-à-vis thirty-four CobiT IT processes [20].

### B. Risk Management in Indian Banking Sector

Over the last couple of decades, India has emerged as one of the fastest growing economies in the world (average 7% GDP growth). India's banking and financial institutions have also experienced high growth rate (18% growth in banking sector) [14]. Rapid stride in ICT (Information and Communication Technology) has helped Indian banking industry to metamorphose from a ledger driven manual activity to a pervasive, fully networked and integrated CBS (Core Banking Services) system. But the undeniable consumer benefits are often offset by techno-procedural complexities in risk management in B2C (Business-to-Consumer) environment. It gives rise to multifarious frauds of alarming proportions [14]. Hence appropriate controls (policies, procedures, guidelines, processes etc. across organizations) need to be implemented to contain the menace.

### C. Compliance Auditing (CA)

As evident from a recent Ernst & Young Global Information Security Survey-2012 [18], stakeholders are increasingly concerned over frauds endangering confidentiality, integrity and security of the organization's information repository. Consequently, organizations are encumbered with the task of synchronizing day to day activities and procedures with a host of standards, regulations and guidelines which are legally binding. CA or Compliance Auditing (which is part and parcel of any regulatory compliance process), formally states whether mandatory controls and safeguards are employed and function correctly. But, without automation, it becomes difficult to manually correlate business practices with conflicting statutory requirements and industry best practices.

## III. LITERATURE SURVEY

In the banking sector, information asymmetry among cooperative/competing agents is one of the root causes of fraud. Anonymity of agents' actions (in different physical and digital channels and payment avenues) results in a state of non-equilibrium of trust and controlling power among interacting agents. Some of these agents try to exploit lacunae in the banking process and technology. This chain of events gives rise to the scope for fraud [16]. Deloitte's fraud survey on Indian Banking sector brings some of these

disturbing trends into the open [14]. According to the survey respondents, "lack of oversight by line managers and/or deviations from existing process/controls" (73%), "current business pressure to meet target" (50%), "difficult business scenarios" (47%), and "lack of automated tools to identify potential red flags" (37%), "collusion between internal staffs and external agencies" (37%) are five major reasons for fraud [14]. All these point to the cardinal importance of automated Compliance Auditing (CA) solutions for information security and mitigation of fraud.

Vendor/technology specific computer-assisted compliance management solutions exist which address a small subset of problems within compliance and primarily focus on lower-level aspects of IT governance such as configuration management, change management, patch management and licensing management [8] [10].

In this paper, we utilized many concepts from diverse fields e.g. fuzzy reasoning system [6] [7] [13] and ontology, which are adopted from ontology engineering [1] [2] and ontology learning [3]. These techniques, along with linguistic tools [4], are used to (semi) automatically extract a body of concepts, relationships and values from various information sources (e.g. employee handbook) to form an ontology. Fuzzy reasoning has also been applied in other important domains such as law, healthcare and financial engineering [5] [11] [13].

## IV. METHODOLOGY

Current research involving creation of ontology based adaptive automated CA system belongs to the design science [19]. Ontology, which is frequently referenced, is "an explicit specification of a shared conceptualization of a domain" [2] [11]. It is constructed to capture implicit, explicit and commonsense-knowledge of a domain such that the knowledge may be shared, accessed, reused and consumed by autonomous computing agents. In this section, we will show how ontology may be used to capture compliance auditing process into reference and target knowledgebase which facilitates organization's performance measurement during auditing.

### A. Ontology of Information Security Concepts

The study will identify a set of notions, viewed as important, in the context of information security. The notions are then defined as concepts. For each concept, the intuitive meaning is documented and the relationships between the concepts are simultaneously derived. The concepts and the relationships are then used to design the ontology. At the same time, any additional attributes, required by the ontological concepts, are categorized. In the next two sub-sections, we divide the conceptual framework in reference and target ontologies.

### B. Reference Ontology

Our application, namely construction of reference ontology, is divided into the following three steps: (i)

Domain Knowledge Modelling (ii) Application Logic Modelling (iii) Application Logic Extension.

(i) Domain Knowledge Modeling: it can be formalized from the following knowledge sources: Information Security Standards and Best Practices (e.g. ISO17799 [17], CobiT4.1 [20]), Information Security Dictionaries (e.g. Glossary for Information Security), Domain Experts Knowledge etc.

(ii) Application Logic Modelling: it can be extracted from the following important knowledge sources: Compliance Requirements from Information Security Standards, Contractual Agreements with Stakeholders, Company Policies (e.g. Employee Handbook) etc.

(iii) Application Logic Extension: it can be formalized from the following knowledge sources: Past compliance results, Business Impact Assessment (BIA), Analysis of financial penalty due to non-compliance etc. Section V will elucidate the entire approach while modelling PO9 of CobiT4.1 [20].

### C. Target Ontology

Target knowledge base is the *Corporate Memory* of the organization. It contains the facts and knowledge about the organization. The knowledge can come from the following sources: Document Management Systems (e.g. knowledge and Meta data), File systems (e.g. instances of documents stored in network drives), Employee Management System (e.g., knowledge of organization entities) etc.

### D. Compliance Rules and Metrics

In order to perform a compliance study, the agent behaviours are to be captured and verified in a particular scenario. Next, it is to be ascertained whether the behavioural trace is consistent with the regulatory requirements. Compliance measure is computed by first converting the compliance rules, expressed in EC (Event Calculus), into a set of high level language like Java. Each node in the tree represents an event (primitive or abstract) spread over an interval of time. Using temporal relations, an event node is related to one or more nodes, at the primitive level.

### E. A Fuzzy Technique for Adaptive Compliance Auditing

In the ideal world, all compliance knowledge and facts are properly captured and stored in an ontology. Thus, machines can confidently reason and infer results based on the precise and complete data. However, this is not the case in the real world, where most of the data is imprecise, incomplete and ambiguous in nature. To capture the imprecision and uncertainty of the real-world knowledge, we make use of Weighted Fuzzy Production Rules (WFPRs), as a mechanism to represent our compliance requirements. We try to mimic the real world scenario of an auditor( who is adjusting and tolerating numerous imprecise or missing compliance data), while (s)he is in the process of concluding whether compliance is achieved or not for the organization. We propose the use of fuzzy logic techniques to address the inherit issues of vagueness and imprecise inferencing in automatic Compliance Auditing[1] [7] [13].

## V. COMPLIANCE MEASUREMENT OF INFORMATION SECURITY STANDARDS

In this section, we closely follow the methodology which was envisaged while constructing reference ontology in Section IV B. Here, we try to model a specific part of a principal security standard viz. CobiT4.1. Implementation of CobiT 4.1 has become mandatory in designing information security in most of the banks in India.

Information, along with systems, networks, hardware, software and supporting processes, are considered valuable assets to any organization. Protection of vulnerable information assets from wide range of threats, mitigation of business risks, ensuring business continuity, maximizing business opportunities etc. may be formally termed as information security. Complete modelling of a particular security standard like CobiT4.1 is beyond the scope of the present paper. Hence, we would like to show the methodology by rendering a part of the CobiT 4.1(Sec. PO9 of CobiT4.1 “Assess and Manage IT Risk”) into ontology based semantic modeling.

a) PO9 from CobiT4.1 establishes an IT risk management framework. It is followed by establishing particular context in which risk assessment framework is applied. Subsequently, specific risky events (events which are capable of producing negative impacts) are identified. Finally, risk response, maintenance and monitoring of risk action plan are undertaken.

b) Recommendations from PO9 from CobiT4.1. are expressed in the following steps:

- Deriving a semantic model (using ontologies from the view point of compliance checking) for information security standard. This model is derived from control statements and auditor’s queries.
- Deriving semantic rules from control statements.
- Applying the rules in the ontology database for checking consistency.
- Deriving strategies for handling partial, incomplete, erroneous and fraudulent data in the ontology.
- Determining the relevance of each concept (using fuzzy weights) for the computation of compliance.
- Computing the compliance measurement.

c) As robust risk management framework is an essential ingredient of Cobit4.1 security regime, the auditor may enquire from the company executives, whether following activities have been properly performed or not.

- i. Performing risk assessment.
- ii. Evaluating strategic/tactical/ business objectives.
- iii. Identifying internal/external critical IT objectives.
- iv. Identifying risk context (Environment, domain, country, regulation, size, etc.).
- v. Identifying events (business oriented, IT related).
- vi. Assessing risk associated with each event (Record & maintain risk registry, cost, benefit etc).

- vii. Identifying strategic risk associated with business (Investment, funding, technology, domain etc.).
- viii. Identifying tactical risk associated with business (Project plans, implementation and other operational issues).
- ix. Selecting, identifying, calculating risk responses.
- x. Prioritising controls for risk mitigation.
- xi. Providing appropriate funding policies in place for risk treatment.
- xii. Maintaining a risk action plan.
- xiii. Performing IT value management (Costs, benefits, Strategy and Tactics).

In Fig. 1, CobiT risk management framework is illustrated for our (partial) source ontology construction. Now, let us critically examine the methodology for the first question (whether or not the company on its part has undertaken proper risk assessment). In Fig. 1, CobiT 4.1 risk management framework is expressed in event based ontology representation. Each box in the Fig. 1 represents an event. The thirteen questions mentioned above may be enquired using structured query language. Here, the model is being represented using Event Calculus (EC). It is a logic formalism used for workflow analysis [22]. In a workflow process represented by EC, there are four major types of activities: (a) sequential activity (b) parallel activity (c) conditional activity and (d) iterative activity [22].

While examining risk assessment in Fig. 1, we work in a bottom-up manner. Setting up risk portfolio is preceded by two sub events (a) identify IT tactical risk and (b) identify IT strategic risk. This can be represented in Event Calculus by AND-join of concurrent activities. The formulation is shown below:

$$\text{happens (end (set risk portfolio), T)} \leftarrow \text{happens (end ((identify IT tactical risk), T1), happens (end ((identify IT strategic risk), T2), T = \max ( T1, T2) )} \quad (1)$$

In a similar manner, risk categorization process can only start when the two sub-processes (setting up risk portfolio and identifying risk trends and events) are over. It is also represented in Event Calculus by AND-join of activities. The Event Calculus formalism is shown below:

$$\text{happens (end (categorize risk), T)} \leftarrow \text{happens (end ((set risk portfolio), T1), happens (end ((identify risk trends and events), T2), T = \max (T1, T2) )} \quad (2)$$

Selecting risk commences sequentially after risk categorization is over and it is represented as:

$$\text{happens (start (select risk), T)} \leftarrow \text{happens (end (categorize risk), T)} \quad (3)$$

Establishing risk context is composed of three sub processes (identifying external and internal context of each risk assessment, selecting risk criteria and selecting goals of risk assessment). These sub processes are also interlinked among themselves. Event Calculus formalism of AND-join activities is shown below:

$$\text{happens (end (establish risk context), T)} \leftarrow \text{happens (end ((identify internal/external context of each risk assessment),$$

$$\text{T1), happens (end ((select risk criteria), T2), happens (end ((select goals of risk assessment), T3), T = \max (T1, T2, T3) )} \quad (4)$$

And, finally, risk assessment is complete after its sub-processes viz. selecting critical IT objectives, establish risk context and selecting risk are complete.

$$\text{happens (end (assessing risk), T)} \leftarrow \text{happens (end ((select critical IT objectives), T1), happens (end ((establish risk context), T2), happens (end ((selecting risk), T3), T = \max (T1, T2, T3) )} \quad (5)$$

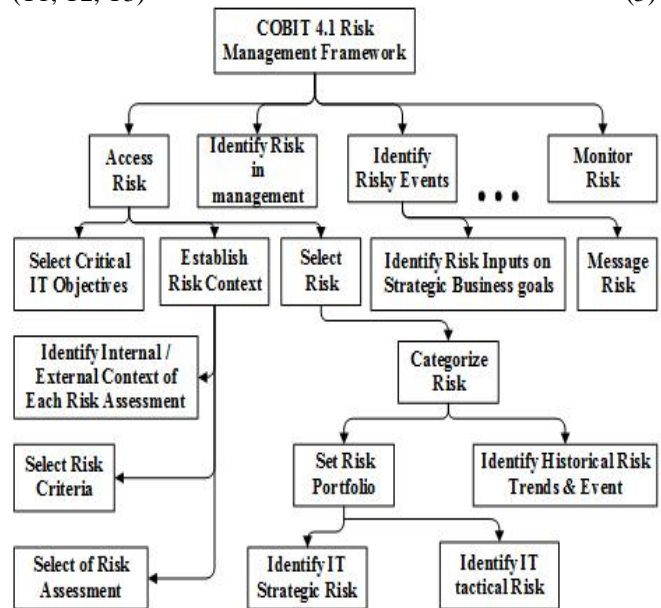


Figure 1. CobiT risk management framework (partial source ontology)

Now, we discuss the same question, (i.e. risk assessment) in a top down manner, in finer detail. We closely examine each of the events in Fig. 1. We try to understand five processes: (i) temporal and causal relationship between events (ii) resource consumption by each event (iii) agents' relationship for each event (iv) the agents' organization associated with each event and (v) objectives of each event. The risk assessment event is composed of three sub events: (a) selecting critical IT objectives (b) establishing risk context and (c) selecting risk. The sub-goals coming in the form of attributes, value and relationship tuple are associated with each event (i) achieving business and IT alignment, priority matching and integration of purposes (ii) assessing business requirement in line with particular enterprise requirement, government regulations, relevant laws and contracts (iii) assessing capabilities and setting of performance metrics in terms of IT's contribution to organizations' goals, objectives, functionality, scalability etc. (iv) setting up of IT strategic and tactical plans (v) performing IT portfolio management by analyzing program portfolios, project and service portfolios (vi) performing IT value management by calculating project costs, benefits, strategy and tactics.



Next, we consider establishing risk context event which is subdivided into three sub-events, namely: identifying internal/external context of each risk assessment, selecting risk criteria and selecting goals of risk assessment. Internal and external risk context is dependent on specific environment like industry, domain, area, country, rules, regulations, best practices, financial health etc. Selecting risk criteria determines organization’s risk tolerance. Selecting goals of risk assessment emphasizes that risk mitigation strategies are encoded within organization culture.

IT strategic risks are concerned with interaction of related stakeholders, strategic matching between IT goals and enterprise vision, investment opportunities, budget and funding, technology maturity etc. The tactical risks are concerned with IT-enabled program investments, IT initiatives in different projects, resource requirements etc. Finally identification of historical risk-trends and events are concerned with setting up of archives of historical cases (it records system failures, that seriously compromised performance).

VI. RISK MANAGEMENT FRAMEWORK OF A BANK X

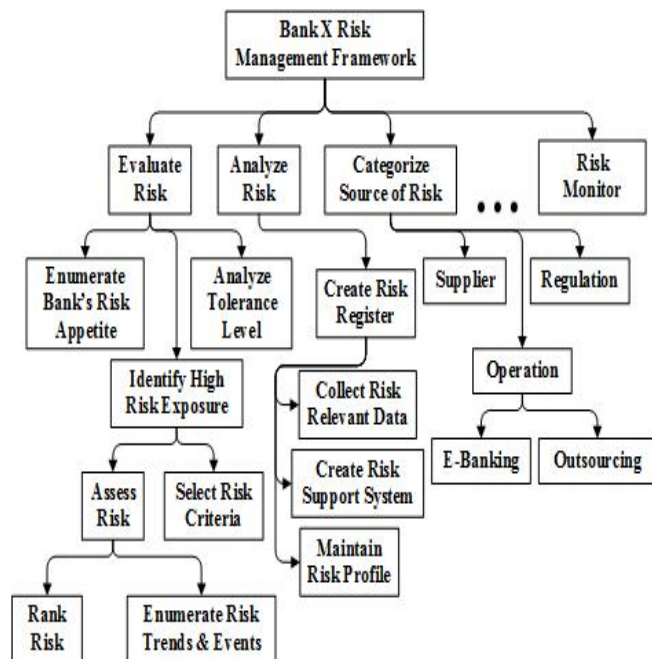


Figure 2. Risk management framework of Bank X (partial target ontology )

In this section, construction of target ontology is envisaged as per guidelines in Section IV C. As per our case study, we are surveying a bank X in India. The bank X has over 1500 branches and offices and is having a total business turnover of more than \$20 billion. Presently, bank X is having a three-tier organizational set-up (consisting of the head office, over 25 regional offices and the branches). Operations of all the branches have been computerized with

ATMs and EFTS (Electronic Fund Transfer System). Risk management framework of the bank X closely follows risk governance framework of Cobit4.1.

The risk management process begins with risk appetite, risk tolerance level and risk exposure of the bank X. Identifying risk, analyzing, evaluating and ranking risk processes, controlling, monitoring and articulating risk impact on information and electronic assets will comprise risk governance of the bank. It is achieved by conducting the following exercises:

- (a) Performing bank X’s enterprise risk assessment.
- (b) Evaluating IT risk tolerance threshold.
- (c) Approving IT risk tolerance.
- (d) Aligning IT risk policy with bank X’s overall risk policy.
- (e) Promoting IT risk consciousness/awareness culture among stakeholders.
- (f) Useful communication of IT risk.
- (g) Approving/accepting IT risk analysis.
- (h) Enriching strategic decision making process using IT risk analysis.
- (i) Prioritizing IT risk response activities.

Risk register or more popularly “risk log” is used for three specific purposes.

- (a) Storing important data for accumulating, identifying, analysing, managing and reporting IT specific risk.
- (b) As part of risk profile maintenance, an up to date inventory of known risk related events and the attributes (disposition, probable impact and expected frequency of occurrence) of IT resources are formed.
- (c) Preparation of decision support system (with respect to risk management framework) is composed of defining / estimating IT risk and identifying risk response options. Categorization of various sources of risk, is of paramount importance in bank X’s risk governance framework.
- (a) Board approved policy, procedure, guidelines.
- (b) Digital signature and evidence (which is taken as legal proof) may be the source of fraud/malpractice.
- (c) Suppliers/contractors are possible sources of risk.
- (d) Adherence to stipulated privacy requirements of customer(s), where the bank is delivering various products/services through electronic banking channels (here the jurisdiction is domain and country specific).
- (e) Granting authorization on need based requirements.
- (f) Monitoring persons with elevated access privilege.
- (g) Appropriate job profiling.
- (h) Evaluating vendors/outsourced services providers (comprehensive due diligence procedures, monitoring performance, managing service-level agreements).
- (i) Operational risk related to e-banking and outsourcing.
- (j) Elimination/restriction on manual intervention for back up, update and data transfer.
- (k) Proper authorization of data in foreign banks having access to bank X’s data.
- (l) Compliance with regulatory, statutory and contractual obligations on the deployment of Information Systems.

Depending on risk appetite of the bank and its impact/significance to the business, bank X management may take recourse to any of the following five actions:

- (a) Ignoring risk (reject risk, if its impact is lower than the risk threshold).
- (b) Avoiding risk (eliminate risk by removing causes).
- (c) Transferring risk (deflect/ allocate/share risk with partners, insurance companies etc.).
- (d) Accepting risk (formal acknowledgement of existence of risk and proper redress).
- (e) Mitigating risk (reduce risk by defining, implementing and monitoring suitable procedures and safeguards).

While mitigating risk, management may choose to either use appropriate controls or reduce risk at acceptable level by using one or many of the following safeguards:

- (a) Detailed inventory control of information and asset.
- (b) Classification of new employees according to risk profile, priority and experience.
- (c) Suitable physical and environmental control.
- (d) Monitoring operational alignment with risk tolerance.
- (e) Awareness training program for employees.
- (f) Setting up robust incidence management process.
- (g) Preparation of detailed audit trail.
- (h) Providing data security measures like cryptography.
- (i) Optimizing system security.
- (j) Checking critical functions (finance, regulation, legal).
- (k) Optimizing response to risk exposure.
- (l) Implementing control for malware protection.
- (m) Robust network protection strategy.
- (n) Strong control for remote computing.

Communication of risk related issues to appropriate forum involves articulation and reaction to risky events.

- (a) IT related loopholes are to be communicated in timely fashion to right forum at right time for right response.
- (b) Immediate gain/loss/opportunities from IT related events are to be exploited.
- (c) Communicating IT risk analysis result.
- (d) Reporting risk management activities and compliances.
- (e) Independent IT assessments are to be interpreted.
- (f) Identifying IT related opportunities.
- (g) Intimating/maintaining incident response plan.

Monitoring IT risk management ensures optimizing & integrating day to day operations with overall IT risk strategy and business decisions.

- (a) Fixing personal or individual accountability for IT risk management.
- (b) Harmonizing business & IT risk strategy.
- (c) Integrating IT risk practices to enterprise risk practice.
- (d) Risk based transaction monitoring surveillance process should be kept in place.
- (e) Optimizing resource allocation for IT risk management
- (f) Independent risk assurance for IT risk management.
- (g) A suitable framework for Business Continuity Management may be implemented.

VII. CALCULATING COMPLIANCE METRIC

In this section, we will show how risk assessment metric can be calculated for CobiT (Fig. 3) to arrive at the compliance measurement. In this tree structure, each node represents weight  $w_i$  which measures relative contribution of each event to the overall event assessing risk. Now, specific weight distribution is beyond the scope of this paper, but nominally they are proportional to fraction of total man-hour an auditor spends in auditing each task. It is arrived after extensive consultation with a number of information system auditors who are engaged in calculating relative weight distribution in risk management of an enterprise undergoing Information Security Compliance. One auditor, while checking for an organization claiming to be CobiT compliant, may first want to verify whether identification of strategic and tactical risk to be .2 and .05, respectively making risk portfolio to be .25 out of .35. The auditor may also give historical risk trends and events to be .1 making categorizing risk to be .35 out of possible .5. The auditor also gives identification of internal/external context of risk assessment a value equal to .15 out of .2.

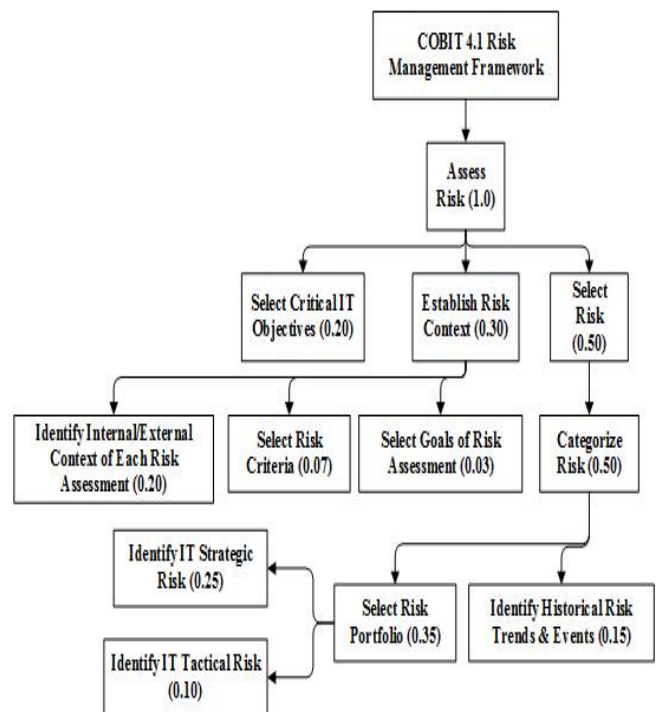


Figure 3. Assessing Risk (CobiT 4.1)

Selecting risk criteria to be .05 and goals of risk assessment to be .01 makes establishing risk context to be .21 out of possible .3. The auditor also selects critical IT objectives to be .15 out of .2. It makes assessing risk to be .71 (.35+.21+.15). Normally, it passes for unqualified opinion from the auditor. In this case, some most critical observation regarding weight distribution and corresponding

evaluation by the auditor may not be out of place. Primarily, two most important cases arise as follows:

(a) Perfect Organization: here, the auditor is satisfied about the procedures followed and data maintained in the organization's database and opine that the organization is following the regulation, and gives an unqualified opinion.

(b) Imperfect Organization: here, the data and procedure maintained by organization leaves much to be desired as per auditing standard. Following sub-cases may arise:

(i) Qualified opinion: the auditor gives an opinion on the organization performance and up-keeping of data and records and methodology used in auditing process ( subject to certain reservations).

(ii) Adverse (Negative) opinion: the auditor determines that he does not agree with the affirmations to be made by the respective organization. Based upon the material facts he may give an adverse opinion on the conduct of the business, resulting in legal, financial problems for the organization.

(iii) Disclaimer of opinions: when an auditor fails to obtain sufficient appropriate audit evidence to warrant an expression of opinion either on the conduct or on the procedure of the business, the auditor may make a disclaimer of opinion on the said organization.

(iv) Piecemeal opinion: such an opinion may be given in case when the auditor concludes that he is hereby unable to give an overall opinion on the statements and procedure of the organization but he can express an opinion limited to certain portion of the audit report of the organization.

In our example, the auditor may give unqualified opinion as the bank compliance touches 71%.

## VIII. CONCLUSION AND FUTURE WORK

In this paper, we proposed an ontology based model for a particular standard (Cobit4.1) in information security domain in Indian banking sector. We will conclude by briefly stating some limitations of our work. While calculating compliance, proper attention was not paid about organization (bank) groupings, culture and size. Due to space constraints, designing architecture of agents performing compliance auditing as well as real life cases involving missing/incomplete/ambiguous data could not be undertaken. For future direction of our research, completeness, redundancy and consistency of analysis of information security model may be undertaken in a formal manner and the comparative studies between different security standards may be investigated in multiple domains.

## REFERENCES

- [1] A. D. Preece, "A new approach to detecting missing knowledge in expert system rule bases", *International Journal of Man-Machine Studies*, Vol. 38, No. 4, pp. 661-668, April, 1993.
- [2] A. Gangemi, "Ontology Design Patterns for Semantic Web Content", In *Proceedings of the 4<sup>th</sup> International Semantic Web Conference (ISWC)*, pp. 262-276, 2005.
- [3] A. Gómez-Pérez, L. M. Fernández, and O. Corcho, "Ontological Engineering with examples from the areas of knowledge management, e-commerce and the semantic web", London: Springer, 2004.
- [4] E. Sanchez, "Fuzzy Logic and the Semantic Web", Elsevier, 2006.
- [5] G. Lau, K. H. Law, and G. Wiederhold, "Legal Information Retrieval and Application to E-Rulemaking", In *Proceedings of the 10<sup>th</sup> International Conference on Artificial Intelligence and Law (ICAIL)*, pp. 146-154, 2005.
- [6] J. Li, "Robust Rule-Based Prediction", *IEEE Tsn on Knowledge and Data Eng.* Vol 18, No. 8, August 2006.
- [7] L. Zadeh, "Fuzzy Sets", *Information and Control*, Vol. 8, pp. 338-353.
- [8] Managesoft, "Managesoft Compliance Manager", (<http://managesoft-compliance-manager.software.informer.com>) (Last access 17/3/13)
- [9] SEC, "Sarbanes Oxley Act 2002", U.S. Securities and Exchange Commission.
- [10] Symantec, "Improving IT Compliance: Guidance for Midsize Organizations", Whitepaper, July, 2006.
- [11] S. Ammar, R. Wright, and S. Selden, "Ranking State Financial Management: A Multilevel Fuzzy Rule-based System", *Decision Sciences* 31 (2), pp. 449-482, 2000.
- [12] US Dept of Health and Human Services, "Health Insurance Portability and Accountability Act (HIPAA)", 1996.
- [13] W. Siler and J. J. Buckley, "Fuzzy Expert Systems and Fuzzy Reasoning", John Wiley & Sons, Inc, 2005.
- [14] Indian Banking Fraud Survey-2012- Navigating the Challenging Environment (Deloitte February 2012).
- [15] Working Group on Information Security, Electronic Banking, Technology Risk Management and Cyber Frauds (RBI, January 2011) ([http://rbidocs.rbi.org.in/rdocs/PublicationReport/Pdfs/WRE\\_B210111.pdf](http://rbidocs.rbi.org.in/rdocs/PublicationReport/Pdfs/WRE_B210111.pdf)) (Last access 17/3/13).
- [16] P. Saha, N. Parameswaran, B.B. Chakraborty, and A. Mahanti" A Formal Analysis of Fraud in Banking Sector" 46th Hawaii International Conference On System Sciences (7-10 January, 2013 Wailea, Maui, HI 96753, USA).
- [17] AS/NZS ISO/IEC 17799:2006 Information Technology — Security Techniques — Code of Practice for Information security Management (July 2006).
- [18] Ernst & Young's 2012 Global Information Security Survey ([http://www.ey.com/Publication/vwLUAssets/Fighting\\_to\\_close\\_the\\_gap:\\_2012\\_Global\\_Information\\_Security\\_Survey/\\$FILE/2012\\_Global\\_Information\\_Security\\_Survey\\_\\_\\_Fightin\\_g\\_to\\_close\\_the\\_gap.pdf](http://www.ey.com/Publication/vwLUAssets/Fighting_to_close_the_gap:_2012_Global_Information_Security_Survey/$FILE/2012_Global_Information_Security_Survey___Fightin_g_to_close_the_gap.pdf)) (Last access 17/3/13).
- [19] A. Hevner, S. March, J. Park, and S. Ram (2004). "Design Science in Information Systems Research", *MIS Quarterly*, Vol. 28, No. 1, pp. 75-105.
- [20] IT Governance Institute: Control Objectives for Information and Related Technologies (COBIT) ([www.itgi.org](http://www.itgi.org)) (Last access 17/3/13)
- [21] Bank for International Settlement (2011) "Basel III: A Global Regulatory Framework for more Resilient Banks and Banking Systems" ([www.bis.org](http://www.bis.org)) (Last access 17/3/13)
- [22] N. K. Cicekli and Y. Yildirim "Formalizing Workflows Using the Event Calculus". *Database and Expert Systems Applications*, Springer 2000.
- [23] K. F. Brickey, "From Enron to Worldcom and Beyond: Life and Crime after Sarbanes-Oxley", *Washington University Law Quarterly* Vol. 81, pp. 357-401, 2003.

# Autonomic Diffusive Load Balancing on Many-core Architecture using Simulated Annealing

Hyunjik Song and Kiyoun Choi  
 School of Electrical Engineering and Computer Science  
 Seoul National University  
 Seoul, Republic of Korea  
 pupasong@dal.snu.ac.kr; kchoi@snu.ac.kr

**Abstract**—Many-core architecture is becoming an attractive design choice in high-end embedded systems design. There are, however, many important design issues, and load balancing is one of them. In this work, we take the approach of diffusive load balancing which enables autonomic load distribution in many-core systems. We modify the existing scheme by adding the concept of simulated annealing for more effective load distribution. The modified scheme is also capable of managing a situation of non-uniform granularity of task loading, which the existing ones cannot. As experiments, we tried various existing schemes as well as the proposed one to map a synthetic application with 30 threads on a many-core architecture with 21 cores and 4 memory tiles. The experiments show that the modified scheme gives results better than the existing approaches.

**Keywords**-diffusive load balancing; simulated annealing.

## I. INTRODUCTION

The rapid increase of semiconductor density has enabled today's high-performance embedded systems to have many-core architecture. However, to utilize maximally the ability of the system, it is very important to map parallel threads properly onto the many-core architecture. There have been numerous studies on this topic, which can be classified into two types: design-time solution and run-time solutions. Design-time solutions determine the mapping during design steps and use the result as a static thread mapping during run time [1]. The major advantage is to remove the overhead of transient thread migration and mapping calculation which run-time counterparts suffer from. The main limitations of design-time solutions are (1) the scalability problem in exploring the entire design space due to the exponential complexity of the mapping problem with respect to the number of parallel threads and (2) lack of adaptivity to changing application behavior. In run-time solutions, known as dynamic load balancing, decentralized methods are favored since centralized ones suffer from the scalability problem (e.g., in gathering global load information and mapping calculation). Our work is based on diffusive load balancing [2][3], which is one of representative decentralized methods for load balancing. Decentralized methods, however, can result in globally gradual load imbalance, which forbids diffusive load balancing to achieve perfect load balancing.

Our idea is to borrow the concept of simulated annealing [4] which has a good characteristic to escape local optimum. When the diffusive load balancing scheme suffers from globally gradual load imbalance, allowing thread migration from under-loaded core to over-loaded core with some probability helps proceed toward the perfect load balancing. The load balancing problem gets more complicated when the loadings of threads are not uniform. Whereas existing diffusive load balancing schemes have little capability to manage such multi-threaded applications, our modified approach can effectively handle the situation. The remainder of this paper is organized as follows. Section II reviews related work and Section III introduces diffusive load balancing in terms of negotiation process. Section IV presents our idea using the concept of simulated annealing. Section V gives experimental results and Section VI concludes the paper.

## II. RELATED WORK

Dynamic load balancing policies are classified into direct and iterative ones. Direct load balancing maps threads onto cores in one step [5][6]. Iterative load balancing maps threads incrementally onto cores by migrating threads to neighbor cores and by repeating the migration steps until equilibrium is reached [2][3][7][8][9]. Direct load balancing methods remove redundant neighbor-to-neighbor thread migrations which iterative methods suffer from. However, direct methods have a significant limitation of high overhead in gathering global information (via core to core communication to exchange load information) and calculating thread mapping based on the global information (running a bin packing algorithm during run time). Iterative methods determine load balancing decisions mostly based on local load information. Thus, the overhead of mapping decision is low. However, the quality of mapping is limited due to the lack of global information and thread migration overhead as explained before. In our work, we aim to obtain perfect or close to perfect load balancing with only local information by adopting the concept of simulated annealing. The approach proposed in [10] also uses the concept of simulated annealing to tackle the diffusive load balancing problem to obtain better load balancing. In this

work, we implement the approach more elaborately and demonstrate its effectiveness by comparing it with various existing diffusive load balancing approaches. In addition, we extend the approach to the case of non-uniform thread loading applications.

### III. RUN-TIME DIFFUSIVE LOAD BALANCING

#### A. Negotiation

Diffusive load balancing tackles load balancing problem by mimicking the physical phenomenon of diffusion. The diffusion forces a system towards a stable (minimum- energy) state with homogeneous distribution (of density of molecules, pressure, etc.). It achieves this by displacing physical objects (mostly, molecules in nature) along the direction of decreasing energy obtained by comparing local states (e.g., local level of concentration or pressure) in a decentralized manner. Diffusive load balancing tries to achieve balanced load distribution in the same way as diffusion in nature. In diffusive load balancing, a thread scheduler running on each core takes a policy of load balancing that is similar to physical laws applied to diffusion. Each core performs load balancing autonomously in a fully decentralized manner as in nature by utilizing only local load information. That is, each core tries to balance the load distribution in a local area. The collective efforts of each core’s load balancing force the global load distribution towards the homogeneous state, i.e., equal load distribution. Each core identifies the state of its load (under-loaded, balanced, or over-loaded) by comparing its own load and that of its neighbors (i.e., local information). If its load state is not balanced, it tries to make it balanced by negotiating with its neighbors on load redistribution, i.e., thread migration. Negotiation is the process to determine, if any, sender and receiver cores to migrate threads. The coverage of neighborhood in negotiation called negotiation coverage (i.e., participants in the negotiation) is one of the most important parameters since it affects the quality of diffusive methods. The case of involving only direct neighbor cores may suffer from lack of global knowledge while too wide a coverage may cause inefficiency due to the increased overhead of load information collection and negotiation. There are four types of negotiation coverage proposed previously: direct neighborhood (DN), average extended neighborhood (AN-d), direct neighborhood repeated (DNR) [2] and direct neighborhood with distorted load information (DND) [3]. In DN, each negotiation covers only two direct neighbor cores. One core initiates the negotiation if it detects load imbalance when comparing its load with that of the other core. Then the initiating core asks the other to balance the load by sending (receiving) thread(s) to (from) the other. The other does not accept the request when the migration reverse the sender/receiver roles of the two cores since, if accepted, it will cause the ping-pong situation where the two cores will continue to exchange the same thread in a ping-pong manner. In AN-d, given a center core, its d-hop

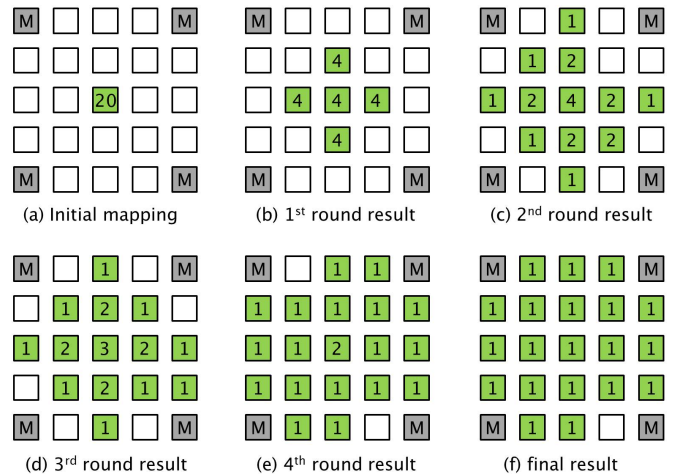


Figure 1. An illustration of diffusive load balancing

neighbor cores participate in the negotiation. Note that as the negotiation coverage increases, the overhead of negotiation increases, so we only consider only 1-hop, i.e., we are only concerned about AN-1(AN) scheme. DNR allows cores to forward a thread from one direct neighbor core to another if there is load difference between them, and DND is the same as DN except that each core gives distorted load information to consider the neighbors’ load status.

#### B. Motivational Example

Fig. 1 shows an example of load distribution obtained by applying the diffusive load balancing to a multi-threaded application on a 21 core architecture. In the figure, a rectangle represents a tile with a core or a memory. The memory tile is annotated with ‘M’. The followings are our assumptions in the example.

- The multi-threaded application with 20 threads has 21 core tiles and four memory tiles.
- Any existing scheme including DN, AN, DNR, or DND can be used as the negotiation coverage and the minimum thread load is ‘1’
- Each thread utilizes one target memory and each memory is utilized by three distinct threads.
- In Fig.1(a), we assume that the entire thread set consisting of 20 threads is initially mapped on a core at (2,2) when the simulation starts ((0, 0) indicates the tile at the upper left corner and (0, 4) indicates the one at the upper right corner).

The threads are mapped onto shaded core tiles in the figure. The number on each shaded core tile represents its load level. Fig. 1(b)-(f) show the results of diffusion rounds.

Note that the diffusion rounds require a large number of intermediate neighbor-to-neighbors thread migrations. Consider the transition from Fig. 1(e) to (f). The core at (2, 2) is over-loaded since its load level is higher than the global

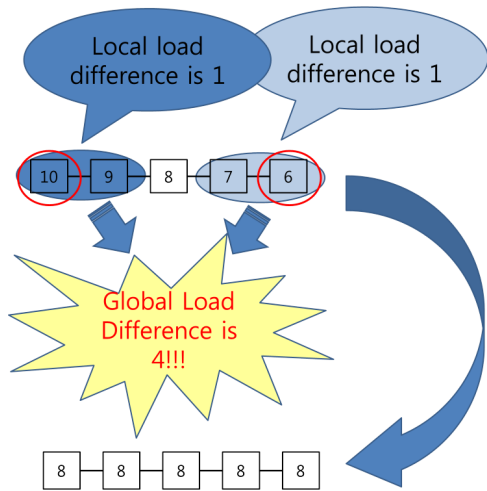


Figure 2. Gradual load imbalance

average load level. On the other hand, the core at (0, 1) and (4, 3) is under-loaded. In the diffusive load balancing with DN as the negotiation coverage, the transition from Fig. 1(e) to (f) cannot take place because any trial to balance the local load with the core at (2, 2) will only change the role of sender and receiver and because the over-loaded core does not know which core is the right receiver due to the lack of global knowledge about under-loaded cores. This is a globally gradual load imbalance problem and unless we have some smart scheme, the transition from Fig. 1(e) to (f) cannot happen. This will be discussed in the next section in more detail.

In terms of performance, global load imbalance can cause significant degradation of the overall system performance. For instance, when a mapping similar to Fig. 1(e) continues, the maximum load '2' determines the total execution time which is 50% longer than the case of ideal load balancing where the maximum load of the 12 cores is '1'.

#### IV. USING SIMULATED ANNEALING

##### A. Globally Gradual Load Imbalance

Diffusive load balancing, when implemented, has a major limitation that it lacks global knowledge. It prevents diffusive load balancing from achieving perfect load balancing. Diffusion of threads, i.e., thread migration is intrinsically based on local load information. Thus, there can be globally gradual load imbalance, since it is difficult to be captured by local information as our motivational example shows in Fig. 2. Even though every adjacent pair of two cores is balanced, global load difference is 4. Sharing global load information among cores could resolve the problem. However, it incurs prohibitively high overhead in continuously collecting the global information from the large number of cores. Thus, the real implementation of diffusive load balancing can fail to achieve perfect load balancing due to the lack of global

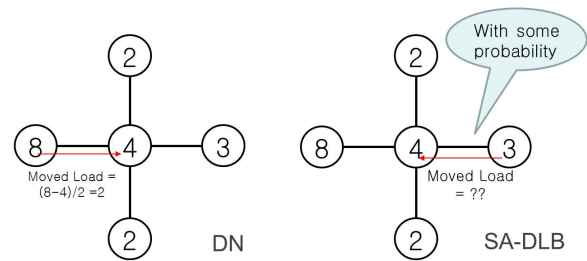


Figure 3. Simulated annealing-based diffusive load balancing

knowledge. It is like a situation that the global optimum cannot be obtained because the solution is not able to escape a local optimum in typical optimization problems. In order to get the global optimum, it is quite effective to exploit metaheuristics such as genetic algorithm, ant-colony optimization, tabu search, and simulated annealing.

##### B. Uniform Load Granularity

Because diffusive load balancing is executed in run-time, applying metaheuristics with huge overhead is not feasible. We accordingly borrow only the concept of simulated annealing, which includes cooling schedule and probability of accepting moves, and so on. The approach in [9] also adopts the concept of simulated annealing for load balancing. However, it does not have the concept of cooling schedule, which will make it difficult to converge to an optimal state. Moreover, there is no experiment given to show the effect of adopting the simulated annealing approach. In the proposed simulated annealing-based diffusive load balancing (SA-DLB) approach, each core performs load balancing based on DN negotiation scheme. The distinguished element of SA-DLB, however, is that an under-loaded core can migrate its threads to an over-loaded one in accordance with a calculated probability, whereas the previous works such as DN, AN, DNR, and DND, etc. do not admit it as shown in Fig. 3. To do so, each core maintains a temperature value internally by lowering the value with a given period. The higher the temperature is, it is more likely for threads to be migrated from the under-loaded core to the over-loaded core, and as the temperature falls down, that kind of migration is observed less frequently. Therefore, the effect of escaping from a local optimum can be seen in the global view, which means there is a good chance of better load balancing. When the statistics of the application loading is constant in time (e.g., the number of threads and the loading of each thread can be modeled as ergodic processes), SA-DLB outperforms other schemes, which has been confirmed by the experiment presented in Section V. A detailed description is shown in Algorithm 1. First, initialize the Temperature value, the most important parameter in simulated annealing. Then, we determine which core will be a sender or receiver. Either an over-loaded core or an under-loaded core can be a sender,

---

**Algorithm 1** SA-based Migration (Uniform Load Granularity)
 

---

```

 $t \leftarrow 0$ 
initialize  $T // \text{Temperature}$ 
repeat
   $r \leftarrow \text{random}(0 \text{ or } 1)$ 
  if  $r = 0$  then
    set over-loaded core as a sender
    and under-loaded core as receiver
  else if  $r = 1$  then
    set under-loaded core as a sender
    and under-loaded core as receiver
  end if
   $\text{load\_gap}_{\text{current}} \leftarrow \text{load}_{\text{sender}} - \text{load}_{\text{receiver}}$ 
   $n \leftarrow \text{random}(0 \dots \text{load}_{\text{sender}})$ 
  for  $i = 0$  to  $n$  do
     $\text{load}_{\text{sender}} \leftarrow \text{load}_{\text{sender}} - 1$ 
     $\text{load}_{\text{receiver}} \leftarrow \text{load}_{\text{receiver}} + 1$ 
  end for
   $\text{load\_gap}_{\text{new}} \leftarrow \text{load}_{\text{sender}} - \text{load}_{\text{receiver}}$ 
   $\text{gain} \leftarrow |\text{load\_gap}_{\text{current}}| - |\text{load\_gap}_{\text{new}}|$ 
  if  $\text{gain} \geq 0$  then
    commit migration
  else
    if  $\text{random}[0, 1) < e^{-\frac{|\text{load\_gap}_{\text{current}}| - |\text{load\_gap}_{\text{new}}|}{kT}}$ 
    then
      commit migration
    end if
  end if
   $T \leftarrow g(T, t)$ 
   $t \leftarrow t + 1$ 
until (halting-criterion)
    
```

---

and the other is set to be a receiver. A sender core generates a random number at most its number of threads. For example, a sender that has three threads can generate 0, 1, 2, or 3. Then the sender calculates the load gap between the sender and the receiver that will be obtained after migrating threads as many as the generated random number. The computed load gap is used to determine whether the probabilistic move will be accepted or not. This process is done iteratively with the temperature value decreasing. In our environment, the temperature value is decreased by 10% at every step of iteration as described with  $g(T, t)$  in Algorithm 1. The halting-criterion is satisfied when the temperature reaches 10% of the initial value.

### C. Non-uniform Load Granularity

As explained in previous subsection, in the SA-DLB method, it is assumed that every thread has the same load size, which is also assumed in the previous approaches such as DN, AN, DND, and DNR. Therefore, the load of each core is calculated as the number of threads assigned to that

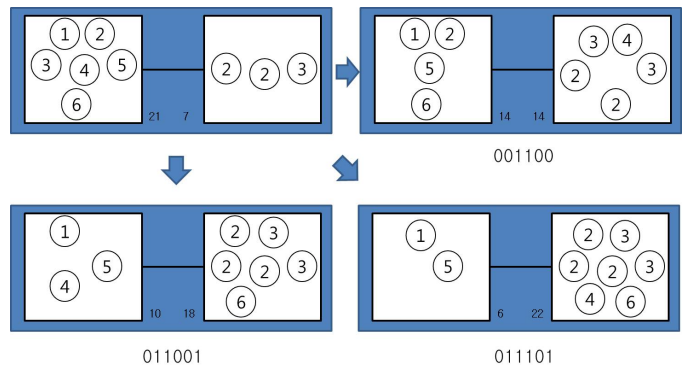


Figure 4. Move generation for non-uniform load situation

core. In this case, any thread can be migrated and it is easy to determine how much loads are to be migrated. For example, to migrate load of amount 10 to a receiver core, the sender core can just send 10 threads. However, the problem becomes more complicated when the application has threads with various kinds of load quantity, i.e., non-uniform load granularity. Because the number of threads does not mean the quantity of load of a core, it is no longer simple to determine how many threads are to be migrated. This implies that it is no more suitable to apply the existing methods such as DN, etc. If we apply SA-DLB, however, we can easily determine which threads are to be migrated and how much load is sent to the target core. Fig. 4 shows the example of move generation of SA-DLB in the situation of non-uniform load granularity. Each white rectangle represents a processor core and each circle represents a thread assigned to the core. Each thread is annotated with its load value and this value is also used as the identifier of the thread (threads with the same load value do not need to be distinguished). In the upper left pair of Fig. 4, one core has six threads with 21 load and the other core has three threads with 7 load. If the DN method is applied to balance the two cores, the left core could send 7 loads to the right one, which makes both cores have the same load, i.e. 14. However, it is not easy to select appropriate threads, so achieving load balance in the global view would be much difficult. Before applying SA-DLB, we assume that the over-loaded core manages the process of load balancing between two cores. We call the managing core a 'controller'. The controller generates random one-bit binary numbers as many as its number of threads. If the generated random numbers are '011001' as in the lower left case of Fig. 4, the threads '2', '3', and '6', which correspond to '1', are regarded as candidates to migrate. If these three threads are moved to the right core, the cores will have 10 and 18 load respectively, and this makes the load difference of the two cores 8, which is smaller than the original gap, 14. In this case, the migration is accepted because the load gap is decreased, implying that the degree of imbalance between two cores is decreased. If

---

**Algorithm 2** SA-based Migration (Non-Uniform Load Granularity)
 

---

```

t ← 0
initialize T // Temperature
repeat
    r ← random(0 or 1)
    if r = 0 then
        set over-loaded core as a sender
        and under-loaded core as receiver
    else if r = 1 then
        set under-loaded core as a sender
        and under-loaded core as receiver
    end if
    load_gap_current ← load_sender - load_receiver
    for i = 0 to number_of_sender_threads do
        rand ← random(0 or 1)
        if rand = 1 then
            load_sender ← load_sender - load_i
            load_receiver ← load_receiver + load_i
        end if
    end for
    load_gap_new ← load_sender - load_receiver
    gain ← |load_gap_current| - |load_gap_new|
    if gain ≥ 0 then
        commit migration
    else
        if random[0, 1) < e $\frac{|load\_gap\_current| - |load\_gap\_new|}{kT}$ 
        then
            commit migration
        end if
    end if
    T ← g(T, t)
    t ← t + 1
until (halting-criterion)
    
```

---

the generated random numbers are '011101' as in the case of lower right in Fig. 4, the resulting gap between the cores is 18 which is bigger than the original value. Even though the imbalance becomes worse, the migration for '011101' is not discarded immediately. Instead, the migration could be accepted with some probability which is shown in Algorithm 2 in detail. For convenience, we call the algorithm a SA-DLB-NU.

## V. EXPERIMENTS

### A. Target Many-Core Architecture

The target many-core architecture consists of 21 ARM946ES core tiles, 4 SRAM memory tiles and 5x5 mesh NoC as shown in Fig.1. The NoC performs XY and worm-hole routing without virtual channel. The NoC router has five ports (one for the local NI, and the other four for neighboring routers). A flit has 8 bytes and a packet has 19 flits. The

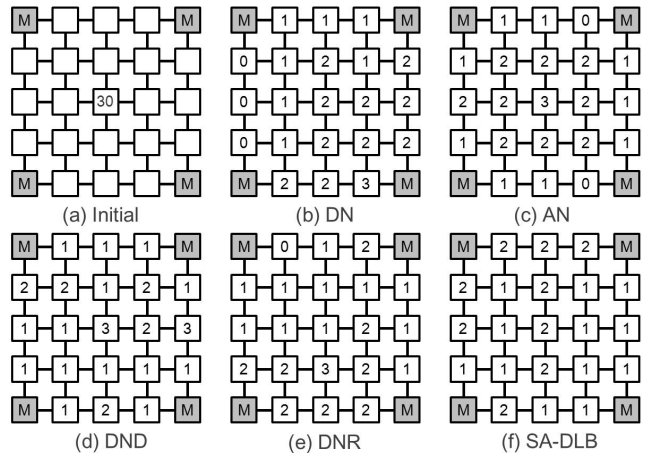


Figure 5. Simulation Results

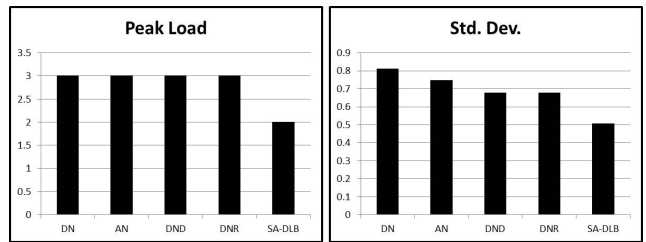


Figure 6. Peak load and standard deviation

NI has a buffer for two packets for pipelined operation. We designed the entire system with a commercial transaction level simulation environment, Carbon SoC Designer [11].

### B. Experimental Results

Fig. 5 shows the results of load distribution obtained by applying the diffusive load balancing to a multi-threaded application on a 21 core architecture (memory tiles are placed at locations different from those in Fig. 1 to see the effects more clearly but the results are not much different). Each white rectangle represents a processor core. The memory tile is annotated with 'M'.

We start with an entire thread set consisting of 30 threads initially mapped on a core at (2, 2) as shown in Fig. 5(a). Simulation results of previous approaches including DN, AN, DND, and DNR are shown in Fig. 5(b)-(e), respectively, together with that of the proposed SA-DLB in Fig. 5(f). In the result of DN, AN, and DNR, threads are well distributed in the sense that for every pair of two neighboring cores the load difference is less than or equal to one. However, we can easily observe global imbalance; in DN scheme, the core at (3, 4) has three threads while cores at (0, 1), (0, 2), and (0, 3) have no thread. Moreover, in the DND scheme, cores at (2, 2) and (1, 2) have a gap of 2. On the other hand, SA-DLB can distribute the thread perfectly as shown in Fig. 5(f). In Fig. 6, the peak value and standard



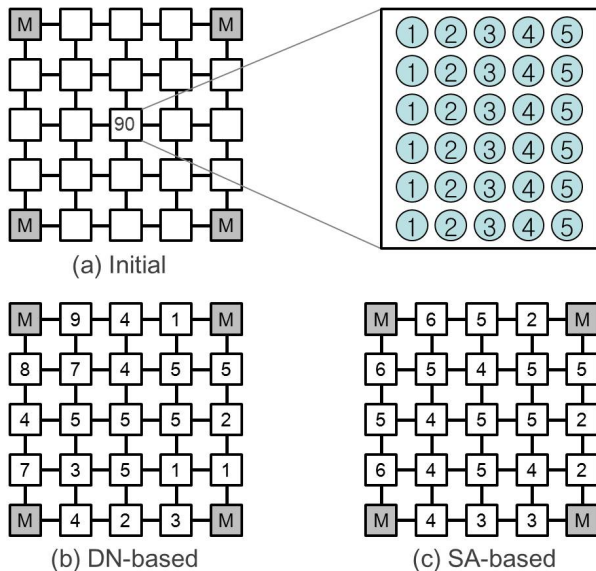


Figure 7. Simulation results

Table I  
SIMULATION RESULTS

	DN-based	SA-based
average	4.28	4.28
standard deviation	2.24	1.27
maximum load	9	6
minimum load	1	2
difference	8	4

deviation of distributed load are shown. Even though the improved version of DN such as AN, DNR, and DND has better standard deviation, our SA-DLB gives the best result in terms of standard deviation. Although SA-DLB provides a better performance for load balancing, it tends to take 3~4 times longer than other approaches, since it has higher computational complexity.

Fig. 7 shows the result of SA-DLB in the case of non-uniform load granularity. Initially, the many-core system has 30 threads with total load amount of 90 on a core at (2, 2). Because there is no previous work that is capable of managing non-uniform cases, we compare our SA-DLB-NU with a DN-like scheme that can handle such cases. The DN-like scheme tries to balance the load with its direct neighbor by migrating threads in a way to make two involving cores to have as same load as possible. Table I summarizes the results shown in Fig. 7. SA-DLB-NU has less standard deviation and the maximum loading is also less than the DN-based diffusion scheme.

## VI. CONCLUSION AND FUTURE WORK

In this contribution, we have introduced a new negotiation scheme for run-time diffusive load balancing on many-core SoC architecture. By adopting the concept of escaping local optimum in simulated annealing, we can distribute

parallel threads more evenly than by using existing negotiation techniques. Our new negotiation scheme can manage the situation of a multi-threaded applications with various thread load granularity. Future work will include making our approach applicable to applications with communicating threads. In addition to the communication of threads, we believe that SA-DLB can also be used for balancing load of applications with dynamically varying statistics.

## ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012-0006272)

## REFERENCES

- [1] Y. Ahn, K. Han, G. Lee, H. Song, J. Yoo, X. Feng, and K. Choi, "Socdal: System-on-chip design accelerator," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 13, no. 1, pp. 17:1–17:38, Feb. 2008.
- [2] A. Corradi, L. Leonardi, and F. Zambonelli, "Diffusive load-balancing policies for dynamic applications," *IEEE Concurrency*, vol. 7, pp. 22–31, Jan. 1999.
- [3] F. Zambonelli, "How to improve local load balancing policies by distorting load information," in *Proceedings of the Fifth International Conference on High Performance Computing*, ser. HiPC '98, Washington, DC, USA, Dec. 1998, pp. 318–339.
- [4] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.
- [5] J. A. Stankovic and I. S. Sidhu, "An adaptive bidding algorithm for processes, clusters and distributed groups," in *ICDCS*, 1984, pp. 49–59.
- [6] M.-Y. Wu, "On runtime parallel scheduling for processor load balancing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 8, no. 2, pp. 173–186, Feb. 1997.
- [7] G. Cybenko, "Dynamic load balancing for distributed memory multiprocessors," *J. Parallel Distrib. Comput.*, vol. 7, no. 2, pp. 279–301, Oct. 1989.
- [8] A. Cortés, A. Ripoll, F. Cedó, M. A. Senar, and E. Luque, "An asynchronous and iterative load balancing algorithm for discrete load model," *J. Parallel Distrib. Comput.*, vol. 62, no. 12, pp. 1729–1746, Dec. 2002.
- [9] E. Jeannot and F. Vernier, "A practical approach of diffusion load balancing algorithms," in *Proceedings of the 12th international conference on Parallel Processing*, ser. Euro-Par'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 211–221.
- [10] H. Song and K. Choi, "Simulated annealing-based diffusive load balancing on many-core soc," in *Proceedings of the 8th ACM international conference on Autonomic computing*, ser. ICAC '11. New York, NY, USA: ACM, 2011, pp. 187–188.
- [11] "Soc designer plus," Jan. 2013. [Online]. Available: <http://www.carbondesignsystems.com>

## Autonomic Ontologies for Governmental Knowledge Base

Stainam Nogueira Brandao  
COPPE/Federal University of Rio de Janeiro  
Rio de Janeiro, Brazil  
stainam@cos.ufrj.br

Tiago Silva  
COPPE/Federal University of Rio de Janeiro  
Rio de Janeiro, Brazil  
tiagoss@cos.ufrj.br

Sergio Assis Rodrigues  
COPPE/Federal University of Rio de Janeiro  
Rio de Janeiro, Brazil  
sergio@cos.ufrj.br

Luis Araujo  
Brazilian Department of the Treasury  
Brasilia, Brazil  
luis.araujo@planejamento.gov.br

Jano Souza  
COPPE/Federal University of Rio de Janeiro  
Rio de Janeiro, Brazil  
jano@cos.ufrj.br

**Abstract**— Ontologies are used as models to represent the semantics of the underlying data. The increasing amount of semantic data brings along important technical challenges for development and maintenance of domain ontologies. Our approach aims to provide the ontologies with capacity to evolve through the follow characteristics: (1) have a direct connection with the real world; (2) be able to execute actions in response to external stimuli; (3) execute actions faster than the human response. In other words, a system with proactive behavior must detect symptoms and must be able to handle such situations without human supervision. The paper describes the governmental knowledge base constructed from Brazilian laws and how it is linked and managed by domain ontologies through the autonomic computing paradigm to implement the proactive behavior. The autonomic characteristics were obtained through architecture that treats ontologies as knowledge that requires a management system to monitor known symptoms and execute specified actions on undesirable scenarios. The existing ontologies in the SIOP-LEGIS [3] repository are currently monitored for symptoms presented in this paper and it reached the ability to recommend actions for domain ontologies' evolution. We envision the autonomic architecture will be able to take actions regarding Service Level Agreement (SLA) and improve the human/system interaction.

**Keywords**— Knowledge base; Linked Data; Autonomic Computing; Ontologies.

### I. INTRODUCTION

According to F.C. Albuquerque et al. [1], Open Government Data integration is possible at a global level promoting the use of standard RDF vocabularies. During the triplification process, adequate tools are thus necessary to help users map local concepts to existing RDF vocabularies, in use by other datasets in the Linked Open Data (LOD) Cloud.

A.G. Silva et al. [8] tackles that classification schemes, such as thesauri or taxonomies, are generally created and

maintained by controlled user groups. Furthermore, several methods have recently been proposed for managing ontologies and knowledge bases. However, as described below, they act on specific activities of ontology engineering.

In our work, the main source for the government knowledge base is the Federal Official Gazette, which is a PDF document and contains legislation, jurisprudence and administrative actions [3]. Published by Authority since 1808, today's Brazilian Gazette is the Brazilian Government's Official Journal. It was set up to provide King John with news while he and his court were in Brazil publishing Decree, Laws, Program and Internal Rules. With a new edition every day, today's Brazilian Gazette contains a huge amount of information and statutory notices about decisions and changes at a local and national level. The Brazilian Gazette is a natural candidate for the Government to semantically enable the reuse potential of the information it contains.

F. Bugiotti et al. [5] assert that the amount of available RDF data sources on the Web increases rapidly and, there is a constant need for scalable RDF data management tools.

Our proposal includes the assessment phase and applies its contribution within knowledge bases that use domain ontologies as semantic resources.

In this paper, autonomic ontologies are the domain ontologies that adhere to a set of active rules that deal with the actions on the configuration, healing, protection and optimization of the ontology.

The rest of this paper is structured as follows. In Section 2, we first define the concepts used in our research. Section 3 describes the related works. Section 4 analyzes the main requirements and Section 5 describes the project for an autonomic system in the context of domain ontologies. In Section 6, we describe a case study of ontologies evolution and management. Section 7 concludes the paper.

II. CONCEPTS AND RELATED TECHNOLOGIES

E.R. Sacramento et al. [13] define and relate ontology, knowledge base and data sources such as used within this research:

- (a) An ontology is a pair  $O=(V,S)$  such that
  - (i)  $V$  is a finite alphabet, the vocabulary of  $O$ , whose atomic concepts and atomic roles are called the classes and properties of  $O$ , respectively, and
  - (ii)  $S$  is a finite set of inclusions in  $V$ , the constraints of  $O$ . The constraints (or Axioms) capture the semantics of the terms.
- (b) A knowledge base is a triple  $KB=(V,S,A)$  such that
  - (i)  $(V,S)$  is an ontology, and
  - (ii)  $A$  is a finite set of assertions in  $V$ .
- (c) A data source is a pair  $DS=(V,A)$  such that
  - (i)  $V$  is a finite alphabet, and
  - (ii)  $A$  is a finite set of assertions in  $V$ .

Similarly, RDF (Resource Description Framework) [17] is a triple subject-property-object, usually described as  $P(S, O)$ , where a given subject  $S$  has a property  $P$  that assumes the value  $O$ . E.R. Sacramento et al. [13] define Linked Data as a set of best practices for publishing and connecting structured data on the Web [18]. From the user's perspective, the main goal of Linked Data is the provision of integrated access to data from a wide range of distributed and heterogeneous data sources [19].

According to F.C. Albuquerque et al. [1], a reactive application advocates a paradigm shifting from human-centered to human-supervised computation. In their perspective, a proactive system must: (1) have a direct connection with the real world; (2) be able to execute actions in response to external stimuli; (3) execute actions faster than the human response. In other words, a system with proactive behavior must detect symptoms and must be able to handle such situations without human supervision. For this, R. Calhau et al. [6] apply technical and administrative procedures for developing, producing and supporting the life cycle of a product to control product evolution.

III. RELATED WORKS

According to M.C.S. Figueroa et al. [16], methodology for building ontologies mainly includes guidelines for single ontology construction ranging from ontology specification to ontology implementation, mainly targeted to ontology researchers. While NeOn Methodology [20], suggests pathways and activities for a variety of scenarios, METHONTOLOGY [21], On-To-Knowledge [22], and DILIGENT [23] were up to 2009 as the most referred methodologies for building ontologies and prescribe a rigid workflow.

A. Knowledge Management

S. Slimani et al. [15] describe distributed ontology evolution approaches, showing that ontology change management increases, especially if services ontologies are heterogeneous (like Semantic Service Architecture - SSOA).

Their approach takes into account some constraints: (1) Ontologies must be autonomous and communicate with each other in reactive way. (2) Not all changes should be managed: there are some changes, which are not interesting to manage because they do not affect the interconnection between ontologies. (3) Ontology should receive just changes that affect the mapping with its interconnected ontologies. (4) One should have a good understanding of changes, that will be translated according to the mapping between ontologies. (5) Mapping is a charred resource between two ontologies and should be managed in parallel since to access to this resource can generate conflicts. The approach is based on a distributed algorithm presenting agent behaviors': (1) initiator ontology agent (IOA) and (2) Dependant Ontology Agent DOA.

B. Proactive System

F.C. Albuquerque et al. [1] discuss basic requirements for proactive real-time monitoring applications. They propose an architecture to deploy applications that monitor moving objects, explore trajectory semantics and are sensitive to environment dynamics. This architecture uses workflows and it features a module to extract data, which helps detect changes on road conditions.

IV. CHOP: DOMAINS ONTOLOGIES WITH AUTONOMIC CHARACTERISTICS

According to M.R. Nami et al. [11], autonomic elements are the heart of an autonomic system. The autonomic elements have a control loop that regulates the workflow of different sub-components of an autonomic system.

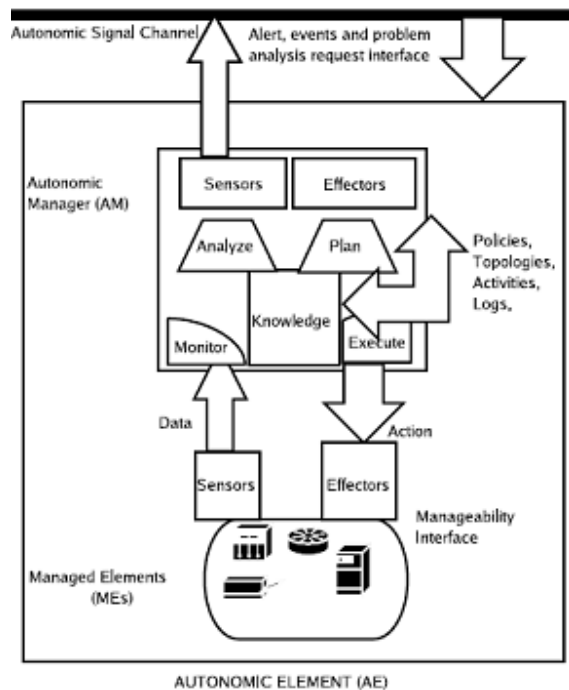


Figure 1. Autonomic Architecture [11]

Figure 1 represents the autonomic project developed by M.R. Nami et al. [11] that provides autonomic features for the domain ontologies, which defines the following components:

- Autonomic Element (AE): basic block of autonomic system, where its interaction with other AE produces the self-managing behavior;
- Managed Element (ME): any resource (in our case, the ontologies) that has its behavior controlled by the environment;
- Autonomic Manager (AM): component that monitors and controls the ME.

Within our approach, each Managed Element (ME) is autonomic domain ontology and the Autonomic Manager (AM) is the meta-knowledge describing the workflow with its specified active rules that are the policies defined by the ontologist.

The autonomic computing paradigm uses actions and predefined rules to lead a new ontology configuration, where the autonomic characteristics of configuration (C) act on ontology for normalization, mapping and alignment to other existing ontologies. Besides, healing actions treat undesirable scenarios during the autonomic evolution. Likewise, ontology instances require actions and rules to address issues related to protection. Also, ontology querying indicates the need for treatment optimization in scenarios that compromise the service quality offered by the ontology.

Then, we define autonomic ontologies as domain ontologies that obey active rules that deal with the special actions on the ontology behavior and their knowledge bases. Accordingly, the actions are related to configuration, healing, protection and optimization of the ontology (Self-CHOP).

## V. AUTONOMIC ACTIONS

According to E. Hovy [9], ontologies are better accepted by traditional critics only if at least two conditions are addressed: they have well-founded methodologies for construction and evaluation and prove their usefulness in real applications. Our proposal contemplates the assessment phase by monitoring ontology metrics and applies its contribution within governmental knowledge bases that use the domain ontology as a semantic resource. Our approach makes use of autonomic computing paradigm to achieve accuracy in the evaluation and ontology management such that the ontologist is spared of the procedure details.

Firstly, the monitoring aims to guarantee the ontology quality with evaluation as an activity of their whole life cycle. This goal is addressed in two scenarios: knowledge base and ontology querying.

### A. Knowledge Base Scenario (Instances)

The knowledge base uses ontologies and vocabularies that already exist and might have been developed by third parties. It is important to monitor and treat events related to these resources interaction. As the knowledge base has

concept's types represented in a domain ontology, we deal with the following events:

#### 1) *New Meanings*

This event occurs when an ontology sub-graph has a concept referenced by a knowledge base and this concept is modified. In this case, the concept instances need to be revised to ensure the real semantic representation between the instance and the modified concept.

#### 2) *Reuse*

This event occurs when the same instance (certified by the same unique identifier or *owl:sameAs* property) exists on different bases and it is from different concepts. In this case, we can infer there is a semantic relationship between these different concepts of ontologies.

#### 3) *Inconsistency*

This event occurs when a concept is deleted. At this point, it is important to identify the sub-graph in which the concept was, as well as, the mappings / integration that this concept had with other ontologies / knowledge base.

### B. Queries Scenarios

Even with the most advanced interfaces for user's interaction, expressing a need for information is a difficult task. There is a semantic distance between the real users needs and what they expressed on the search. The queries performed on ontology provide statistics about its use as a resource semantic related to data quality and needs for ontological management. This scenario includes three events:

#### 1) *Concepts Accessed*

The architecture monitors central ontology concepts to collect data for statistical redistribution of instances. First, SPARQL queries [14] received from client applications are processed to analyse and identify the instances type retrieved from queries through the *rdf type property*. After, the more the concept is quoted, the more it fits in the central ontology concepts group. This means that the concept is quoted when its instances are implicitly mentioned in the query.

#### 2) *Critical Path*

The event occurs identifying the ontology's sub-graph with the largest execution times of queries. From this point, extracting the concepts involved in the SPARQL query [14], class attributes and modifiers used, in our case, order by, projection, distinct, offset and limit (known area of database).

#### 3) *Denial of Service*

Event identified when overload or ineffectiveness access to ontologies. Ontology as a knowledge representation and semantic resource for querying by other systems, must be concerned with the service quality offered and, most importantly, if the service is actually being offered. The two events above address quality while this event verifies availability, keeping the service history offered.

The metric used is the response time of queries to client applications, when they reach the maximum waiting time defined by the ontologist. As shown in the previous event, every query has its runtime recorded and when it achieves an unacceptable level, this event is triggered.

C. Autonomic actions

Autonomic Action is any algorithm developed under autonomic computing paradigm that acts upon domain ontologies in order to generate a new configuration, healing, protection or optimization. The autonomic action is performed after an event is identified.

1) *Balancing Semantic Action*

This action includes or maps a concept to an unbalanced sub-graph through common instances between the sub-graph concepts and concepts from other ontologies. The advantage is to guarantee ontologies mapping, since the common instance ratifies the semantic relation between the concepts involved. Thus, the approach aims to restore the ontology in a coordinated and orderly way to avoid unexpected / unwanted results, maintaining consistency based on metrics already established in the literature.

As the structure taxonomic metrics evaluate the ontology quality structure, the guard expressions use the Width and Depth metrics to identify a sub-graph that reaches a value not desirable by ontologist.

This action treats the problem of sub-graph by mapping concepts with common instances. When the guard expression is triggered, the instances associated with the sub-graph concepts are used as input for the re-design of the unbalanced sub-graph through the following algorithm:

- a) Identification of the sub-graph;
- b) Sub-graph analysis;

c) If the sub-graph has reached a non-acceptable value for the width, then the treatment action will be vertical with the identification of 'NEW concepts' with semantic relation with child classes of the sub-graph (Figure 2);

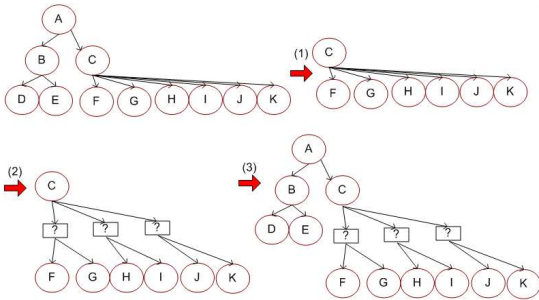


Figure 2. Semantic Vertical Balancing Strategy

d) If the sub-graph has reached a non-acceptable value for depth, then the treatment will use the concept of inclusion on leaf of the sub-graph between a father-class and child-class, expanding the ontology vertically (Figure 3).

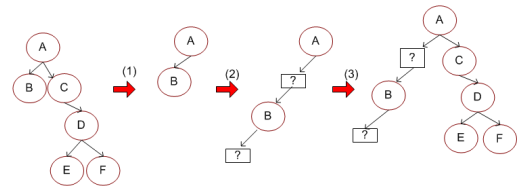


Figure 3. Semantic Horizontal Balancing Strategy

- e) Retrieve the sub-graph instances;
- f) For each instance:
  - g) Check if the resource (identified by rdf:resource) is referenced by other instances, even if in other ontologies;
  - h) Check if the concept type is different between instances identified in the previous step;
  - i) Check the existence of a common concept between the concept type of instances;
  - j) Create a semantic relationship between the concepts identified in the previous step;

2) *Fragmentation Action*

Fragmentation action occurs on the concept being highly referenced by instances and other concepts. This action finds equivalent classes to heal the critical path of ontology. This is possible through (1) equivalent relationship between instances of different ontologies (by *sameIndividuals axiom*) and (2) different instances reference the same resource (by *rdf:resource property*).

Thus, the approach heals the critical path with inclusion of existing concepts to avoid overload in query performance. The increase and enrichment of knowledge bases are the source for healing of ontologies referenced by them.

Fragmentation action has guard expressions associated with the following metrics: Importance of Class (instances distribution), Wealth of classes (instances distribution between classes) and Cost Based Evaluation (CBE - to measure performance).

When any of the ontology metrics reaches a value that triggers at least one guard expressions, Fragmentation action is performed:

- a) Identification of the concept;
- b) Instances selection of the concept identified on step (a);
- c) For each instance:
  - d) Identify instances (1) that reference the same resource (by rdf:resource tag) or (2) has the sameIndividuals axiom with an instance that has a different type (rdf type property) (according to Figure 4);
  - e) Check if the instance type identified (by rdf: type property) is different from the selected instance;
  - f) Inclusion of the equivalentClass axiom between the concept identified in step (a) and the concept of the instance identified in the step (d);

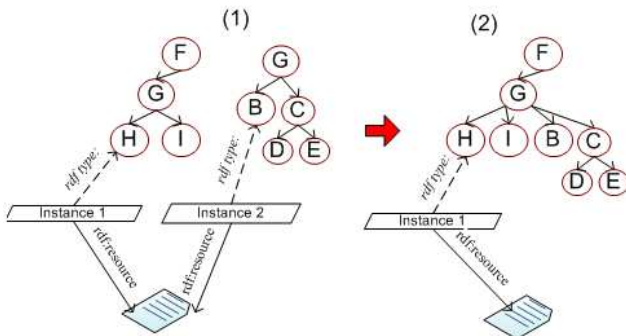


Figure 4. In (2) a relationship was created between the concepts H and B due to their particular instances to represent the same feature (1)

Figure 4 shows the case where two individuals of different types represent the same resource. In this case, they are considered identical individuals according to our approach.

VI. CASE STUDY

Domain ontologies are Managed Elements (ME), in which the metrics are monitored in the form of Jess production rules (Figure 5) [10], implementing the workflow transition conditions of each ontology management. Moreover, the ontology is registered as a web service, whose desirable values are filled by the ontologist. At this moment begins the self-management.

```
(defrule INSTANCES_OVERLOAD
  ?o <- (OntologyVO (e_av_cnt_c ?e_av_cnt_c) (e_max_cnt_c ?e_max_cnt_c > (* e_av_cnt_c 3)))
  =>
  (save-problem "PROTECTION" "INSTANCES_OVERLOAD")
  (printout t "Self-protection -> e_av_cnt_c: " ?e_av_cnt_c
    " e_max_cnt_c: " ?e_max_cnt_c crlf)
  )
```

Figure 5. Jess production rules

This section presents a brief case study to demonstrate how the architecture works. The case came from Secretary of Federal Treasury responsible for control and oversight of federal spending in accordance with the legislation, case law and administrative acts. The Knowledge Organizational System - SIOP-LEGIS [3] is a project that aims to provide knowledge management for legislative domain through changeable representation, which deals with trends in the law.

Nowadays, according to S.N. Brandao et al.[25], the system represents the knowledge from Official Gazette, allowing to answer questions that were required during the monitoring, auditing and oversight. This knowledge base is linked to other Brazilian Open Data and represents one more effort in Open Government Partnership [12] to reflect the country's commitment to strengthen the transparency of government actions to prevent and combat corruption [4].

The government knowledge base is constructed from Brazilian laws, the events that surround them and authorities responsible for these. The knowledge base is built on RDF language through the Brazilian Official Gazette, which is the access for official information.

The initial study was done with two different ontologies: Social Security and Legislative domain [7]. The first ontology has an overload concept called 'Law' (Figure 6), while, Legislative ontology used by Chamber of Deputies treat specifically the federal law documents with other 14 concepts (Figure 7). Given the need to deal with the overloaded concept 'Law' on Social Security ontology, which reference the same official documents of Legislative ontology, which more specific types allowing to treat the overload 'Law' concept in the Social Security ontology. The Fragmentation action maps the Legislative ontology that provides new concepts as view that is a faithful and attentive to changes in the ontology provider. Note that there is a copy or a mapping of new concepts.

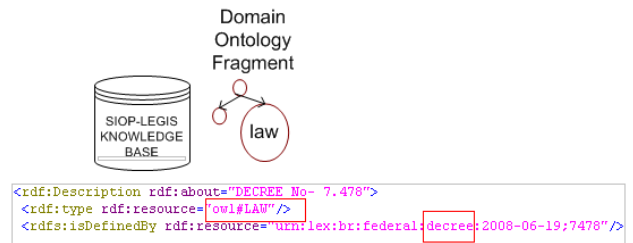


Figure 6. An knowledge base instance associated to concept 'Law' trough 'rdf: type' despite being a decree

A second study was conducted with the inclusion of the concept 'Law' in the Legislative ontology. In this case, the Balancing Semantic action could be performed with the inclusion of hierarchical relationship (containing 14 concepts) in the Social Security ontology. In this case, even the fragmentation action being thrown to the inclusion of the axiom equivalentclass, there was the possibility of performing semantic balancing action.

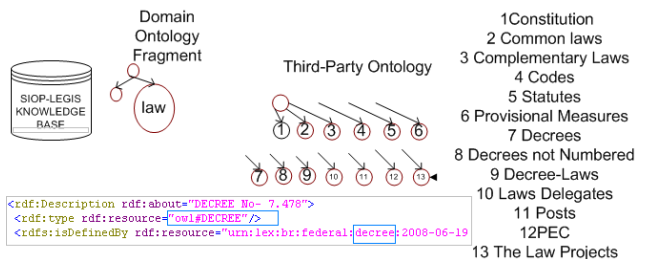


Figure 7. After Fragmentation action, the instance of the Figure 6 now associated to concept 'Decree' trough 'rdf: type', what really represents the 'rdfs: isDefinedBy' property

This suggests as future work to create a workflow with actions containing priority, treatment of infinite loop and treatment of undesirable behavior and scenarios.

VII. CONCLUSION

The SIOP-LEGIS [3] project is part of Federal Budget Secretary initiative to provide information to society. The project allows the development of tools to read the data provided by own government, since information is linked and interoperable in our open knowledge base.

In general terms, the methodology presents as main advantages: (i) the semi-automation process of domain ontologies management, minimizing human intervention, (ii) ontology monitor through ontology metrics, since the knowledge base is constantly updated and consequently under failures. Therefore, if an known symptoms occurs, the proposal allows a new configuration, healing, optimization or protection. The existing ontologies in the SIOP-LEGIS [3] repository are currently monitored for symptoms and it reached the predictive level 3 (according to Figure 8) with the ability to monitor symptoms and recommend actions as a form of domain ontologies' evolution. We envision the next autonomic level, where the architecture will be able to take actions regarding Service Level Agreement (SLA) and improve the human/system interaction.

	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4
Characteristics	Multiple sources of system generated data	Consolidation of data and actions through management tools	System monitors, correlates and recommends actions	System monitors, correlates and takes action
Skills	Requires extensive, highly skilled IT staff	IT staff analyzes and takes action	IT staff approves and initiates actions	IT staff manages performance against SLAs
Benefits		Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency
Manual				

Figure 8. Autonomic level define by [24]

We also aim to find inconsistencies in the knowledge base and indicate them for the domain specialist. With this, self-management characteristics will be added to the knowledge base.

ACKNOWLEDGMENT

We thank CAPES, CNPq and Brazilian Department of Treasury for supporting this work.

REFERENCES

[1] F.C. Albuquerque, I. Barbosa, M.A. Casanova, M.T.M. Carvalho, and J.A.F. Macedo, "Proactive Monitoring of Moving Objects", ICEIS, Wroclaw, Poland, 2012, pp. 191-194.

[2] K. Breitman, P. Salas, M.A. Casanova, D. Saraiva, and M. Chaves, "Open government data in Brazil", IEEE Intelligent Systems, 27(3), Rio de Janeiro, Brazil, 2012, doi: {10.1109/MIS.2012.25}, pp.45 -49.

[3] S.N. Brandao, T.S. Silva, S.A. Rodrigues, L.S. Araujo, and J.M. Souza, "SIOP-LEGIS: Thesaurus for Selection and Management of Brazilian Treasury Domain", International Conference on Knowledge Management and Information Sharing, Paris, France, 2011, pp. 195-200.

[4] S.N. Brandao, T.S. Silva, S.A. Rodrigues, L.S. Araujo, and J.M. Souza, "Knowledge Representation of Brazilian Official Gazettes for Chronological Recovery of Laws", 10° IADIS International Conference WWW/INTERNET, Rio de Janeiro, Brazil, 2011, pp. 149-157.

[5] F. Bugiotti, F. Goasdoué, Z. Kaoudi, and I. Manolescu, "RDF data management in the Amatic cloud", Proceedings of the 2012 Joint EDBT/ICDT Workshops. EDBT-ICDT '12. New York, USA: ACM, pp. 61-72.

[6] R. Calhau and R.A. Falbo, "A Configuration Management Task Ontology for Semantic Integration". Proceedings of the 27th Annual ACM Symposium on Applied Computing. SAC '12. New York, USA, 2012, pp. 348-353.

[7] BCD, Legislação. Available at: <http://www4.planalto.gov.br/legislacao> [retrieved: February, 2013].

[8] A.G. Silva, O.A. Corcho, H. Alani, and A.G. Pérez, "Review of the State of the Art: Discovering and Associating Semantics to Tags in Folksonomies", Knowl. Eng. Rev., 27(1), 2012, doi: 10.1017/S02698891100018X, pp.57-85.

[9] E. Hovy, "Methodologies for the Reliable Construction of Ontological Knowledge", Eds. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 91-106.

[10] Jess, "Jess: Rule Engine and Scripting Environment", Available at: <http://www.jessrules.com/jess/docs/index.shtml> . [retrieved: February, 2013]

[11] M.R. Nami and K. Bertels, "A Survey of Autonomic Computing Systems", In Proceedings of the Third International Conference on Autonomic and Autonomous Systems. ICAS 2007. Washington, DC, USA: IEEE Computer Society, pp. 1-26.

[12] OGP, Open Government Partnership, Available at: <http://www.opengovpartnership.org/countries/brazil>[retrieved: February, 2013].

[13] E.R. Sacramento, M.A. Casanova, K.K. Breitman, A.L Furtado, and V.M.P. Vidal, "Dealing with Inconsistencies in Linked Data Mashups", Proceedings of the 16th International Database Engineering &#38; Applications Symposium. IDEAS '2012. New York, USA: ACM, pp. 175-180.

[14] E. Sirin and B. Parsia, "SPARQL-DL: SPARQL Query for OWL-DL", 3rd OWL Experiences and Directions Workshop (OWLED-2007), Innsbruck, Austria, pp. 16-21.

[15] S. Slimani, S. Baïna, and K. Baïna, "A Framework for Ontology Evolution Management in SSOA-Based Systems", Proceedings of the 2011 IEEE International Conference on Web Services (ICWS). Washington, DC, USA, pp. 724-725.

[16] M.C.S. Figueroa, R.G. Castro, B.V. Terrazas, and A.G. Pérez, "Essentials In Ontology Engineering: Methodologies, Languages, and Tools", Proceedings of the 2nd Workshop Organized by the EEB Data Models Community, Sophia Antipolis, France, 2011, pp.9-21.

[17] F. Manola and E. Miller, 2004. "RDF Primer". Available at: <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>. [retrieved: February, 2013].

[18] C. Bizer, R. Cyganiak, and T. Heath, "How to Publish Linked Data on the Web", Available at: <http://www4.wiwiss.fuberlin.de/bizer/pub/LinkedDataTutorial/> . [retrieved: February, 2013].

- [19] C. Bizer, T Heath, and T.B. Lee, "Linked Data – The Story So Far", *International Journal on Semantic Web and Information Systems*, 5(3), 2009, pp.1–22, doi:10.4018/jswis.2009081901.
- [20] A.G. Perez and M.C.S. Figueroa, "Scenarios for building ontology networks within the NeOn methodology", *Proceedings of the fifth international conference on Knowledge capture. K-CAP '2009*. New York, USA, pp. 183–184.
- [21] M.F. Lopez, A.G. Perez and, N. Juristo, "METHONTOLOGY: from Ontological Art towards Ontological Engineering", *Proceedings of the AAAI 1997 Spring Symposium*. Stanford, USA, pp. 33–40.
- [22] C. Fluit, H.T. Horst, J.V.D. Meer, C.D. Fensel, and E. Ab, "On-To-Knowledge: Content-driven Knowledge management Tools through Evolving Ontologies On-To-Knowledge Consortium", 2000, Available at: <http://www.ontotext.com/research/otk>. [retrieved: February, 2013]
- [23] H.S. Pinto, S. Staab, C. Tempich, Y. Sure, "Distributed Engineering of Ontologies (DILIGENT)", in: Staab, S., Stuckenschmidt, H. (Eds.), *Semantic Web and Peer-to-Peer*. Springer Berlin Heidelberg, 2006, ISBN 978-3-540-28347-8, pp. 303–322.
- [24] IBM, IBM Research | Almaden Research Center | Computer Science [WWW Document]. Available at: <http://www.almaden.ibm.com/cs/projects/autonomic/>. [retrieved: February, 2013]
- [25] S.N. Brandao, S.A. Rodrigues, T.S. Silva, L.S. Araujo, and J.M. Souza, "Open Government Knowledge Base". *The Seventh International Conference on Digital Society*, Nice, France, ISBN: 978-1-61208-249-3, pp. 13-19.