CLOUD COMPUTING 2015

The Sixth International Conference on Cloud Computing, GRIDs, and Virtualization

ISBN: 978-1-61208-388-9

March 22 - 27, 2015

Nice, France

CLOUD COMPUTING 2015 Editors

Yong Woo Lee, University of Seoul, Korea

Carlos Becker Westphall, Federal University of Santa Catarina, Brazil

CLOUD COMPUTING 2015

Forward

The Sixth International Conference on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING 2015), held between March 22-27, 2015 in Nice, France, continued a series of events targeted to prospect the applications supported by the new paradigm and validate the techniques and the mechanisms. A complementary target was to identify the open issues and the challenges to fix them, especially on security, privacy, and inter- and intra-clouds protocols.

Cloud computing is a normal evolution of distributed computing combined with Service-oriented architecture, leveraging most of the GRID features and Virtualization merits. The technology foundations for cloud computing led to a new approach of reusing what was achieved in GRID computing with support from virtualization.

The conference had the following tracks:

- Cloud computing
- Computing in virtualization-based environments
- Platforms, infrastructures and applications
- Challenging features

Similar to the previous edition, this event attracted excellent contributions and active participation from all over the world. We were very pleased to receive top quality contributions.

We take here the opportunity to warmly thank all the members of the CLOUD COMPUTING 2015 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 that dedicated much of their time and effort to contribute to CLOUD COMPUTING 2015. 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 also gratefully thank the members of the CLOUD COMPUTING 2015 organizing committee for their help in handling the logistics and for their work that made this professional meeting a success.

We hope that CLOUD COMPUTING 2015 was a successful international forum for the exchange of ideas and results between academia and industry and to promote further progress in the area of cloud computing, GRIDs and virtualization. We also hope that Nice, France provided a

pleasant environment during the conference and everyone saved some time to enjoy the charm of the city.

CLOUD COMPUTING 2015 Chairs

CLOUD COMPUTING 2015 Advisory Chairs

Jaime Lloret Mauri, Polytechnic University of Valencia, Spain
Wolf Zimmermann, Martin-Luther University Halle-Wittenberg, Germany
Yong Woo Lee, University of Seoul, Korea
Alain April, École de Technologie Supérieure - Montreal, Canada
Aaron McConnell, University of Ulster, UK
Christoph Reich, Furtwangen University, Germany

CLOUD COMPUTING 2015 Industry/Research Chairs

Wolfgang Gentzsch, The UberCloud, Germany
Tony Shan, Keane Inc., USA
Donglin Xia, Microsoft Corporation, USA
Laura Moore, SAP, UK
Anna Schwanengel, Siemens AG, Germany
Atsuji Sekiguchi, Fujitsu Laboratories Ltd., Japan

COULD COMPUTING 2015 Special Area Chairs

Security
Aljosa Pasic, ATOS Research, Spain
Chih-Cheng Hung, Southern Polytechnic State University - Marietta, USA

GRID
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Javier Diaz-Montes, Rutgers University, USA
Nam Beng Tan, Nanyang Polytechnic, Singapore

Autonomic computing
Ivan Rodero, Rutgers the State University of New Jersey/NSF Center for Autonomic Computing, USA
Hong Zhu, Oxford Brookes University, UK

Service-oriented
Qi Yu, Rochester Institute of Technology, USA

Platforms

Arden Agopyan, ClouadArena, Turkey
Dariusz Król, Academic Computer Center CYFRONET - Cracow, Poland

CLOUD COMPUTING 2015

Committee

CLOUD COMPUTING Advisory Chairs

Jaime Lloret Mauri, Polytechnic University of Valencia, Spain
Wolf Zimmermann, Martin-Luther University Halle-Wittenberg, Germany
Yong Woo Lee, University of Seoul, Korea
Alain April, École de Technologie Supérieure - Montreal, Canada
Aaron McConnell, University of Ulster, UK
Christoph Reich, Furtwangen University, Germany

CLOUD COMPUTING 2015 Industry/Research Chairs

Wolfgang Gentzsch, The UberCloud, Germany
Tony Shan, Keane Inc., USA
Donglin Xia, Microsoft Corporation, USA
Laura Moore, SAP, UK
Anna Schwanengel, Siemens AG, Germany
Atsuji Sekiguchi, Fujitsu Laboratories Ltd., Japan

COULD COMPUTING 2015 Special Area Chairs

Security
Aljosa Pasic, ATOS Research, Spain
Chih-Cheng Hung, Southern Polytechnic State University - Marietta, USA

GRID
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Javier Diaz-Montes, Rutgers University, USA
Nam Beng Tan, Nanyang Polytechnic, Singapore

Autonomic computing
Ivan Rodero, Rutgers the State University of New Jersey/NSF Center for Autonomic Computing, USA
Hong Zhu, Oxford Brookes University, UK

Service-oriented
Qi Yu, Rochester Institute of Technology, USA

Platforms
Arden Agopyan, ClouadArena, Turkey
Dariusz Król, Academic Computer Center CYFRONET - Cracow, Poland

CLOUD COMPUTING 2015 Technical Program Committee

Jemal Abawajy, Deakin University - Victoria, Australia
Imad Abbadi, University of Oxford, UK
Taher M. Ali, Gulf University for Science & Technology, Kuwait
Alain April, École de Technologie Supérieure - Montreal, Canada
Alvaro E. Arenas, Instituto de Empresa Business School, Spain
José Enrique Armendáriz-Iñigo, Public University of Navarre, Spain
Irina Astrova, Tallinn University of Technology, Estonia
Benjamin Aziz, University of Portsmouth, UK
Xiaoying Bai, Tsinghua University, China
Panagiotis Bamidis, Aristotle University of Thessaloniki, Greece
Luis Eduardo Bautista Villalpando, Autonomous University of Aguascalientes, Mexico
Carlos Becker Westphall, Federal University of Santa Catarina, Brazil
Ali Beklen, CloudArena, Turkey
Elhadj Benkhelifa, Staffordshire University, UK
Andreas Berl, University of Passau, Germany
Simona Bernardi, Centro Universitario de la Defensa / Academia General Militar - Zaragoza, Spain
Nik Bessis, University of Derby, UK
Peter Charles Bloodsworth, National University of Sciences and Technology (NUST), Pakistan
Ranieri Baraglia, Istituto di Scienza e Tecnologie dell'Informazione - Consiglio Nazionale delle Ricerche, Italy
Alexander Bolotov, University of Westminster, UK
Sara Bouchenak, University of Grenoble I, France
William Buchanan, Edinburgh Napier University, UK
Ali R. Butt, Virginia Tech, USA
Massimo Cafaro, University of Salento, Italy
Mustafa Canim, IBM Thomas J. Watson Research Center, USA
Massimo Canonico, University of Piemonte Orientale, Italy
Paolo Campegiani, University of Rome Tor Vergata, Italy
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Carmen Carrión Espinosa, Universidad de Castilla-La Mancha, Spain
Simon Caton, Karlsruhe Institute of Technology, Germany
K. Chandrasekaran, National Institute of Technology Karnataka, India
Hsi-Ya Chang, National Center for High-Performance Computing (NCHC), Taiwan
Rong N Chang, IBM T.J. Watson Research Center, USA
Ruay-Shiung Chang, Taiwan Hospitality and Tourism College, Taiwan
Kyle Chard, University of Chicago and Argonne National Laboratory, USA
Antonin Chazalet, IT&Labs, France

Nils Grushka, NEC Laboratories Europe - Heidelberg, Germany
Jordi Guitart, Universitat Politècnica de Catalunya - Barcelona Tech, Spain
Marjan Gusev, “Ss. Cyril and Methodius” University of Skopje, Macedonia
Jordi Guitart, Universitat Politècnica de Catalunya - Barcelona Tech, Spain
Yi-Ke Guo, Imperial College London, UK
Marjan Gushev, Univ. Sts Cyril and Methodius, Macedonia
Rui Han, Institute of Computing Technology - Chinese Academy of Sciences, China
Weili Han, Fudan University, China
Haiwu He, INRIA, France
Sergio Hernández, University of Zaragoza, Spain
Neil Chue Hong, University of Edinburgh, UK
Pao-Ann Hsiung, National Chung Cheng University, Taiwan
Lei Huang, Prairie View A&M University, USA
Chih-Cheng Hung, Southern Polytechnic State University - Marietta, USA
Richard Hill, University of Derby, UK
Uwe Hohenstein, Siemens AG, Germany
Luigi Lo Iacono, Cologne University of Applied Sciences, Germany
Shadi Ibrahim, INRIA Rennes - Bretagne Atlantique Research Center, France
Yoshiro Imai, Kagawa University, Japan
Anca Daniela Ionita, University "Politehnica" of Bucharest, Romania
Xuxian Jiang, North Carolina State University, USA
Eugene John, The University of Texas at San Antonio, USA
Foued Jrad, KIT - Universität des Landes Baden-Württemberg, Germany
Carlos Juiz, Universitat de les Illes Balears, Spain
Verena Kantere, University of Geneva, Switzerland
Sokratis K. Katsikas, University of Piraeus, Greece
Takis Katsoulakos, INLECOM Systems, UK
Zaheer Khan, University of the West of England, UK
Prashant Khanna, JK Lakshmipat University, Jaipur, India
Shinji Kikuchi, Fujitsu Laboratories Ltd., Japan
Peter Kilpatrick, Queen's University Belfast, UK
Tan Kok Kiong, National University of Singapore, Singapore
William Knottenbelt, Imperial College London - South Kensington Campus, UK
Sinan Kockara, University of Central Arkansas, USA
Joanna Kolodziej, University of Bielsko-Biala, Poland
Kenji Kono, Keio University, Japan
Dimitri Konstantas, University of Geneva, Switzerland
Arne Koschel, Hochschule Hannover, Germany
George Kousiouris, National Technical University of Athens, Greece
Sotiris Koussouris, National Technical University of Athens, Greece
Kenichi Kourai, Kyushu Institute of Technology, Japan
Nane Kratzke, Lübeck University of Applied Sciences, Germany
Heinz Kredel, Universität Mannheim, Germany
Dariusz Król, Academic Computer Center CYFRONET - Cracow, Poland

Hans Günther Kruse, Universität Mannheim, Germany
Eric Kuada, Aalborg University - Copenhagen, Denmark
Alex Kuo, University of Victoria, Canada
Tobias Kurze, Karlsruher Institut für Technologie (KIT), Germany
Dharmender Singh Kushwaha, Motilal Nehru National Institute of Technology - Allahabad, India
Dimosthenis Kyriazis, University of Piraeus, Greece
Giuseppe La Torre, University of Catania, Italy
Romain Laborde, University Paul Sabatier, France
Petros Lampsas, Central Greece University of Applied Sciences, Greece
Erwin Laure, KTH, Sweden
Alexander Lazovik, University of Groningen, The Netherlands
Craig Lee, The Aerospace Corporation, USA
Yong Woo Lee, University of Seoul. Korea
Grace Lewis, CMU Software Engineering Institute - Pittsburgh, USA
Jianxin Li, Beihang University, China
Kuan-Ching Li, Providence University, Taiwan
Maik A. Lindner, SAP Labs, LLC. - Palo Alto, USA
Maozhen Li, Brunel University - Uxbridge, UK
Dan Lin, Missouri University of Science and Technology, USA
Panos Linos, Butler University, USA
Xiaoqing (Frank) Liu, Missouri University of Science and Technology, USA
Xiaodong Liu, Edinburgh Napier University, UK
Xumin Liu, Rochester Institute of Technology, USA
H. Karen Lu, CISSP/Gemalto, Inc., USA
Glenn R Luecke, Iowa State University, USA
Mon-Yen Luo, National Kaohsiung University of Applied Sciences, Taiwan
Ilias Maglogiannis, University of Central Greece - Lamia, Greece
Rabi N. Mahapatra, Texas A&M University, USA
Shikharesh Majumdar, Carleton University, Canada
Olivier Markowitch, Universite Libre de Bruxelles, Belgium
Ming Mao, University of Virginia, USA
Attila Csaba Marosi, MTA SZTAKI Computer and Automation Research Institute/Hungarian Academy of Sciences - Budapest, Hungary
Gregorio Martinez, University of Murcia, Spain
Goran Martinovic, J.J. Strossmayer University of Osijek, Croatia
Philippe Massonet, CETIC, Belgium
Michael Maurer, Vienna University of Technology, Austria
Reinhard Mayer, Universität Heidelberg, Germany
Aaron McConnell, University of Ulster Coleraine, UK
Per Håkon Meland, SINTEF ICT, Norway
Jean-Marc Menaud, Mines Nantes, France
Andreas Menychtas, National Technical University of Athens, Greece
Jose Merseguer, Universidad de Zaragoza, Spain
Shigeru Miyake, Hitachi Ltd., Japan

Mohamed Mohamed, IBM US Almaden, USA
Owen Molloy, National University of Ireland – Galway, Ireland
Paolo Mori, Istituto di Informatica e Telematica (IIT) - Consiglio Nazionale delle Ricerche (CNR), Italy
Louise Moser, University of California - Santa Barbara, USA
Claude Moulin, Technology University of Compiègne, France
Francesc D. Muñoz-Escoí, Universitat Politècnica de València, Spain
Hidemoto Nakada, National Institute of Advanced Industrial Science and Technology (AIST), Japan
Rich Neill, Cablevision Systems, USA
Surya Nepal, CSIRO ICT Centre, Australia
Rodrigo Neves Calheiros, University of Melbourne, Australia
Toan Nguyen, INRIA, France
Bogdan Nicolae, IBM Research, Ireland
Aida Omerovic, SINTEF, Norway
Ammar Oulamara, University of Lorraine, France
Alexander Paar, TWT GmbH Science and Innovation, Germany
Claus Pahl, Dublin City University, Ireland
Brajendra Panda, University of Arkansas, USA
Massimo Paolucci, DOCOMO Labs, Italy
Alexander Papaspyrou, Technische Universität Dortmund, Germany
Valerio Pascucci, University of Utah, USA
Aljosa Pasic, Atos Research, Spain
David Paul, University of Newcastle, Australia
Al-Sakib Khan Pathan, International Islamic University Malaysia (IIUM), Malaysia
Siani Pearson, Hewlett-Packard Laboratories, USA
Antonio J. Peña, Mathematics and Computer Science Division - Argonne National Laboratory, USA
Giovanna Petrone, University of Torino, Italy
Sabri Pllana, University of Vienna, Austria
Agostino Poggi, Università degli Studi di Parma, Italy
Jari Porras, Lappeenranta University of Technology, Finland
Thomas E. Potok, Oak Ridge National Laboratory, USA
Francesco Quaglia, Sapienza Univesita' di Roma, Italy
Xinyu Que, IBM T.J. Watson Researcher Center, USA
Manuel Ramos Cabrer, University of Vigo, Spain
Rajendra K. Raj, Rochester Institute of Technology, USA
Christoph Reich, Hochschule Furtwangen University, Germany
Dolores Rexachs, University Autonoma of Barcelona (UAB), Spain
Sebastian Rieger, University of Applied Sciences Fulda, Germany
Sashko Ristov, “Ss. Cyril and Methodius” University of Skopje, Macedonia
Norbert Ritter, University of Hamburg, Germany
Ivan Rodero, Rutgers University, USA
Daniel A. Rodríguez Silva, Galician Research and Development Center in Advanced

Telecomunications" (GRADIANT), Spain
Paolo Romano, Instituto Superior Técnico/INESC-ID Lisbon, Portugal
Thomas Rübsamen, Furtwangen University, Germany
Kyung Dong Ryu, IBM T.J. Watson Research Center, USA
Majd F. Sakr, Carnegie Mellon University in Qatar, Qatar
Hadi Salimi, Iran University of Science and Technology - Tehran, Iran
Altino Sampaio, Instituto Politécnico do Porto, Portugal
Iñigo San Aniceto Orbegozo, Universidad Complutense de Madrid, Spain
Elena Sanchez Nielsen, Universidad de La Laguna, Spain
Volker Sander, FH Aachen University of Applied Sciences, Germany
Gregor Schiele, Digital Enterprise Research Institute (DERI) at the National University of Ireland, Galway (NUIG), Ireland
Alexander Schill, Technische Universität Dresden, Germany
Michael Schumacher-Debril, Institut Informatique de Gestion - HES-SO, Switzerland
Wael Sellami, Faculty of Economic Sciences and Management of Sfax, Tunisia
Hermes Senger, Federal University of Sao Carlos, Brazil
Larry Weidong Shi, University of Houston, USA
Alan Sill, Texas Tech University, USA
Fernando Silva Parreiras, FUMEC University, Brazil
Luca Silvestri, University of Rome "Tor Vergata", Italy
Alex Sim, Lawrence Berkeley National Laboratory, USA
Luca Spalazzi, Università Politecnica delle Marche - Ancona, Italy
George Spanoudakis, City University London, UK
Rizou Stamatia, Singular Logic S.A., Greece
Katerina Stamou, Institute of Services Science - University of Geneva, Switzerland
Marco Aurelio Stelmar Netto, IBM Research, Brazil
Hung-Min Sun, National Tsing Hua University, Taiwan
Yasuyuki Tahara, University of Electro-Communications, Japan
Ibrahim Takouna, Hasso Plattner Institute - University of Postdam, Germany
Jie Tao, Steinbuch Centre for Computing/Karlsruhe Institute of Technology (KIT), Germany
Joe M. Tekli, Lebanese American University, Lebanon
Orazio Tomarchio, University of Catania, Italy
Stefano Travelli, Entaksi Solutions Srl, Italy
Parimala Thulasiraman, University of Manitoba, Canada
Ruppa Thulasiram, University of Manitoba, Canada
Raul Valin, Swansea University, UK
Geoffroy R. Vallee, Oak Ridge National Laboratory, USA
Luis Vaquero, HP Labs Bristol, UK
Michael Vassilakopoulos, University of Thessaly, Greece
Jose Luis Vazquez-Poletti, Universidad Complutense de Madrid, Spain
Luís Veiga, Instituto Superior Técnico - ULisboa / INESC-ID Lisboa, Portugal
Salvatore Venticinque, Second University of Naples - Aversa, Italy
Mario Jose Villamizar Cano, Universidad de loa Andes - Bogotá, Colombia
Salvatore Vitabile, University of Palermo, Italy

Bruno Volckaert, Ghent University - iMinds, Belgium
Eugen Volk, High Performance Computing Center Stuttgart (HLRS) - Stuttgart, Germany
Lizhe Wang, Center for Earth Observation & Digital Earth - Chinese Academy of Sciences, China
Zhi Wang, Florida State University, USA
Mandy Weißbach, University of Halle, Germany
Philipp Wieder, Gesellschaft fuer wissenschaftliche Datenverarbeitung mbH - Goettingen (GWDG), Germany
John Williams, Massachusetts Institute of Technology, USA
Peter Wong, SDL Fredhopper, Netherlands
Christos Xenakis, University of Piraeus, Greece
Hiroshi Yamada, Tokyo University of Agriculture and Technology, Japan
Chao-Tung Yang, Tunghai University, Taiwan R.O.C.
Hongji Yang, De Montfort University (DMU) - Leicester, UK
Yanjiang Yang, Institute for Infocomm Research, Singapore
Ustun Yildiz, University of California, USA
Qi Yu, Rochester Institute of Technology, USA
Jong P. Yoon, Mercy College - Dobbs Ferry, USA
Jie Yu, National University of Defense Technology (NUDT), China
Ze Yu, University of Florida, USA
Massimo Villari, University of Messina, Italy
Vadim Zaliva, Tristero Consulting / Carnegie Mellon University (CMU), USA
Baokang Zhao, National University of Defence Technology, China
Xinghui Zhao, Washington State University Vancouver, Canada
Zibin (Ben) Zheng, Shenzhen Research Institute, The Chinese University of Hong Kong, Hong Kong
Jingyu Zhou, Shanghai Jiao Tong University, China
Hong Zhu, Oxford Brookes University, UK
Wolf Zimmermann, University of Halle, Germany

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 or 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 is 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

Nane Kratzke

Identification of IT Security and Legal Requirements Regarding Cloud Services

Constantin Christmann, Jürgen Falkner, Andrea Horch, Holger Kett
Fraunhofer Institute for Industrial Engineering IAO
Stuttgart, Germany
e-mail: {constantin.christmann, juergen.falkner, andrea.horch, holger.kett}@iao.fraunhofer.de

***Abstract*—The adoption of cloud computing holds tremendous potential for small and medium-sized enterprises (SMEs) as it enables them to reduce costs as well as improve flexibility and scalability. In order to choose an appropriate cloud service, it is necessary to carefully identify functional and non-functional requirements. In this regard, the aspects affecting IT security as well as legal requirements are important topics to the users of SMEs. However, the smaller the enterprise, the lower the probability that there is enough expertise to identify the requirements in these subject areas. Addressing this issue, the contribution of this paper is the description of a method for identifying IT security and legal requirements regarding a cloud service in a structured kind of way. The presented method was implemented as a part of a prototype for a cloud service search. Based on this search system, an evaluation with users of SMEs was conducted. The evaluation results attest the search system and its underlying method to successfully assist users regarding the identification of relevant legal and IT security requirements, hence reducing the amount of expertise required by users of SMEs, as well as the associated effort of searching for appropriate cloud services.**

Keywords- cloud computing; cloud services; IT security; legal; requirments; service search.

I. INTRODUCTION

The adoption of cloud computing holds tremendous potential for organizations of all shapes and sizes. Especially small and medium-sized enterprises (SMEs) are highly interested in cloud computing as it enables them to reduce costs as well as improve flexibility and scalability [1]. In order to choose the right cloud service, a thorough identification of functional as well as non-functional requirements regarding such service is necessary. Due to the necessary outsourcing of data – which is often a crucial asset of an enterprise - aspects of cloud services affecting information technology (IT) security as well as legal requirements are important topics to users of SMEs [2][3]. However, the smaller the enterprise, the lower the probability that there is enough expertise to identify the requirements in these subject areas. Hence, the contribution of this paper is the description of a method for identifying IT security and legal requirements regarding a cloud service in a structured kind of way. We did implement this method as part of a prototype for a cloud service search (depicted in Fig. 1). In this search system, the search is based on functional and non-functional requirements. In order to define the non-functional requirements – addressing the subject areas of IT security as well as legal aspects of cloud computing - the user is asked simple questions which are easy to answer without special expertise in both subject areas. These answers serve as an input to the method presented in this paper and the result is a selection of applicable requirements addressing IT security or legal constraints. These requirements then serve as input to a service search which accesses a repository to select appropriate cloud services matching the functional and non-functional requirements. The results of this search are then displayed to the user, making sure that the user will consider only cloud services which satisfy her needs.

The remainder of the paper is structured as follows: Section II gives an overview over the different subject areas of cloud service requirements and justifies the focus of this paper. Section III covers the related work regarding the identification of IT security and legal requirements. Then, in Section IV, the underlying methodology for the development of the presented method is described, and in Section V, the method is explained in detail. Section VI presents the application of the presented method in a website helping users to identify appropriate cloud services. Section VII describes our evaluation efforts and Section VIII concludes with a discussion and an outlook on further research activities.

II. CLOUD SERVICE REQUIREMENTS AND FOCUS

In the field of software engineering, the requirements regarding a software application are typically categorized into functional and non-functional requirements [4]. This schema can also be used for the categorization of requirements regarding a cloud service:

- **Functional requirements:** This category comprises the functions/modules needed by users of the cloud service. Examples are address administration, e-mail or invoice practice. The needed interfaces to other services or applications also fall into this category.
- **Non-functional requirements:** Non-functional requirements typically cover various *qualitative and quantitative aspects* regarding the software (i.e., usability, performance, IT security or documentation). However, for an adequate selection of cloud services we need a broader understanding of non-functional requirements which – along with qualitative aspects – also considers the following:

 ISBN: 978-1-61208-388-9

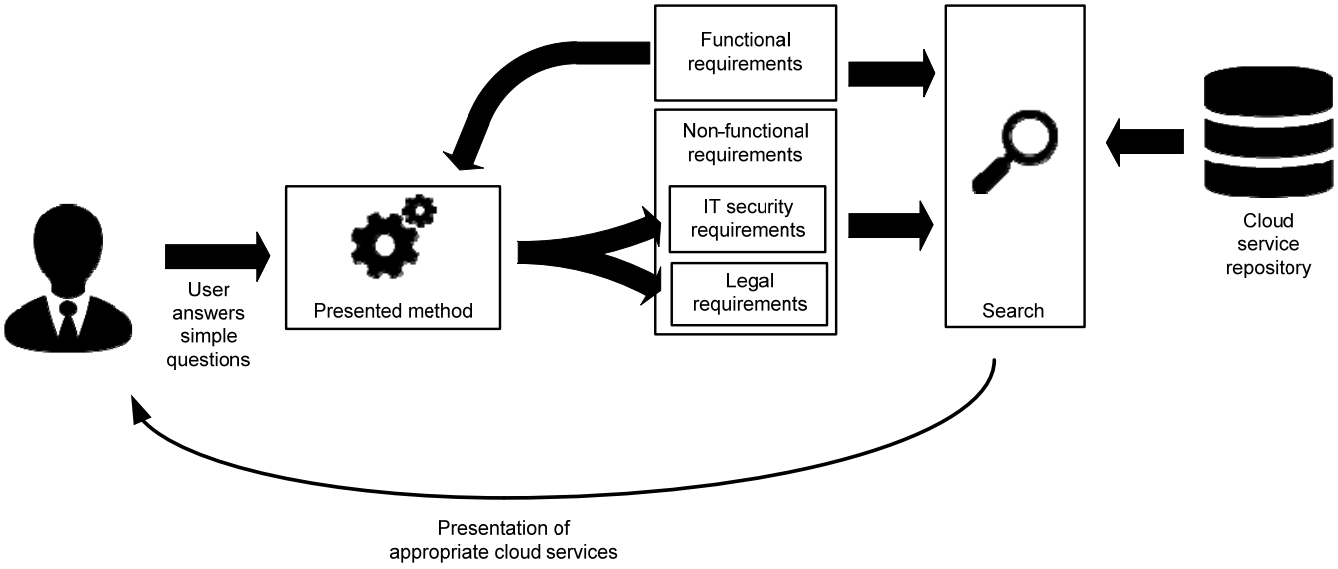


Figure 1. Visualization of the cloud service search

- **Legal requirements:** Depending on the kind of data that is processed by the cloud service, it might be necessary to obey certain legal constraints – i.e., in Germany the Federal Data Protection Act (Bundesdatenschutz-gesetz) regulates the processing of personal data. Such legal requirements may influence (improve) also the IT security of a service; however, primarily they are of a regulatory nature - i.e., they may ensure that fiscal authorities can access service data easily.
- **Economic aspects:** When selecting a cloud service, it might be necessary to also consider economic aspects – i.e., the price of the service should not exceed a given budget or migration cost to a service must be taken into account.

Fig. 2 shows the taxonomy which forms the basis for our understanding of cloud service requirements.

In order to select an appropriate cloud service, all aspects need to be taken into account. However, users from SMEs require varying support in the different subject areas. For example, for reviewing the user experience of certain cloud software not much technical expertise is required. Also, the evaluation of a cloud service from an economic perspective is something users of SMEs should be capable of. However, studies show that such users are especially concerned with IT security as well as being compliant to applicable law [2][3] and these are the topics the users of SMEs in general do not have much experience with. As IT security and law are the areas the users of SMEs need the most support, the focus of this paper lies on the identification of requirements out of these two subject areas.

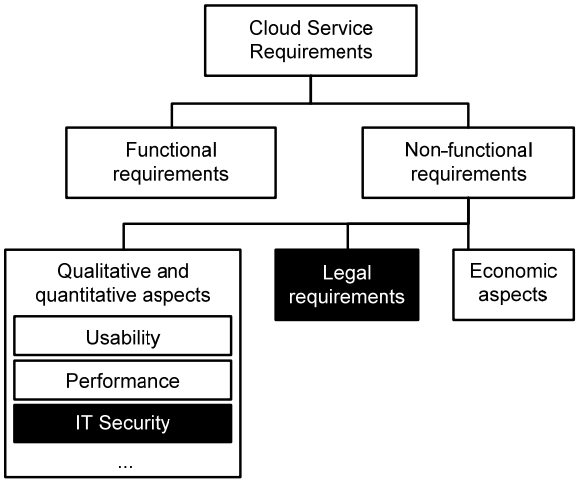


Figure 2. Taxonomy of cloud service requirements. In the diagram the focus areas of this paper has been highlighted.

III. Related Work

As IT security is a highly relevant topic in the context of cloud computing, there exists various literature supporting providers and users to obey important security aspects of cloud computing and cloud services. For example Mather et al. [5] and Krutz et al. [6] cover security aspects of cloud computing mostly with an emphasis on the enterprise perspective. Another example are the security recommendations for cloud providers [7] published by the German Federal Office for Information Security, which can also act as a source for requirements for users of cloud services. On an European Union level the European Union Agency for Network and Information Security (ENISA) offers thread analysis for internet and cloud architectures [8] and, with members from all over the world, the Cloud

 ISBN: 978-1-61208-388-9

Security Alliance (CSA) promotes the use of best practices for providing security assurance on cloud computing for an even broader audience [9]. Furthermore, various certificates formulate requirements affecting the IT security and can be applied in the context of cloud services [10]. Examples are the certification regarding ISO 27001 [11], which addresses IT security management in general or the Euro Cloud Star Audit [12], which explicitly addresses cloud services.

The legal aspects of cloud computing/cloud services do – especially in Germany and the EU as a whole – get great attention and various publications cover relevant aspects for providers as well as users of cloud services [13][14][15].

To sum up, in both areas - IT security as well as law - many resources are available, which can be used by users of cloud computing to formulate their requirements regarding a cloud service. However, to do so, a certain expertise is necessary and in both fields no approach exists, which helps users of cloud services to identify the appropriate requirements without requiring the user to deal with the sometimes rather complicated details.

Recommender systems seek to predict a rating or preference of a user for a specific item. There exist different recommender systems which support the selection of appropriate cloud services [16][17]; however, existing systems mainly focus on quantitative aspects like execution time, response time or budget – hence, leaving out important qualitative aspects with respect to legal constraints or IT security. Furthermore, in these systems the user has to specify the (quantitative) requirements herself which may not be possible for many users of SMEs.

IV. METHODOLOGY

This section covers the individual steps which led to the development of the presented method and its prototypical application. Fig. 3 gives an overview of the different steps involved in this process.

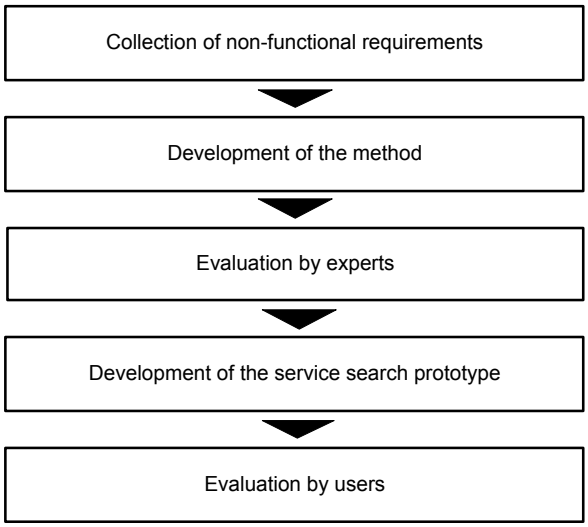


Figure 3. Overview of the different steps which led to the presented method and its prototypical application.

At first, literature reviews in the field of IT security and regarding legal aspects of cloud computing were conducted. Based on [7], [10], [14], [22], [23], and [24] a comprehensive list of non-functional requirements was collected – all of them being relevant regarding the selection of cloud services. The final list of requirements contains 104 entities each being assigned to one of the following subject areas (for examples, see Section V):

- Data center: Infrastructural aspects relevant to IT security, processes, organization of staff (35 requirements)
- Service provider: processes, organization of staff (17 requirements)
- Cloud service: Various aspects of the service relevant to IT security, training and support (35 requirements)
- Legal: contract, legal requirements (17 requirements)

In a next step, strategies regarding IT risk assessment were investigated. As the foundation for our method, we chose the concept of risk analysis [19], which combines the damage a security breach could do to the IT system of a user/organization with its associated probability for occurrence. Then, the list of non-functional requirements as well as the developed method were improved based on discussions with experts with expertise in IT security and regarding legal aspects of cloud computing. The final method is presented in Section V of this paper.

This method was then implemented as part of a prototype for a web-based search system for SMEs which supports the selection of adequate cloud services (see Section VI). The last step comprised an evaluation of this web-based prototype by users from SMEs. The evaluation setting and the results of both evaluation stages (experts and users) are described in Section VII.

V. DESCRIPTION OF THE METHOD

Fig. 4 visualizes the data flow giving an overview over the individual steps of the presented method. The initial input is the set of functional requirements of the user regarding a potential cloud service. The first step performs a preprocessing which results in data types and legal constraints due to the functional requirements. Based on the data types, the protection needs to become qualified. Afterwards, the individual non-functional requirements are derived taking into account these protection needs as well as the legal constraints. The final result is a set of individual non-functional requirements regarding a cloud service. In the following, the different steps of the method will be described in detail.

 ISBN: 978-1-61208-388-9

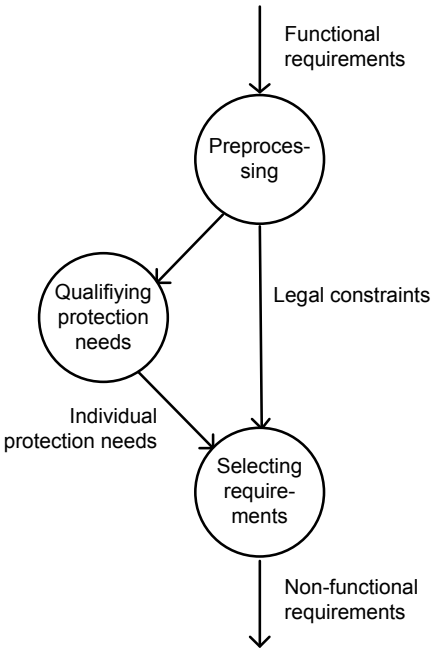


Figure 4. Visualization of the data flow

A. Preprocessing

In the first step a preprocessing is performed, where, based on the functional requirements, the following information is derived:

- The different **data types** which should be processed by a cloud service. As an example, if a functional requirement for the cloud service is *address administration*, then this implies the processing of *address data*.
- The **legal constraints** regarding the cloud service due to the processing of the different data types. These constraints must be derived from national word of law and from regulations applicable to the user's business. Example: In the case of *address data* being a data type, in Germany, the processing of personal data has a legal constraint – due to the fact that the regulations of the Federal Data Protection Act have to be obeyed.

B. Qualifying Protection Needs

Regarding IT security, the most prominent protection targets are availability, integrity and confidentiality of data [18]. In order to qualify the individual user needs regarding these protection targets, our method utilizes the concept of risk analysis [19], which combines the damage a security breach could have to the IT system of a user/organization with its associated probability for occurrence. The risk analysis used by our method comprises the following steps:

1. **Value of protection:** For each data type and protection target (availability, integrity, confidentiality) the value of the protection is qualified using the categories *low, normal, high, very high*. Then, following the maximum principle [20] the value of protecting a target is chosen as the maximum value over all data types.
2. **Threat characteristic:** Then a *threat characteristic* matching the industry sector must be chosen. Based on this threat characteristic, the probability of a security breach affecting one of the protection targets is qualified using the categories *very rare, rare, occasionally, often, very often* (see Table I).
3. **Individual protection needs:** Based on the value of protection and the probability of a security breach, the individual needs regarding the protection targets are derived as being *low, medium, high* or *very high* (see Table II).

TABLE I. THREAT CHARACTERISTIC

Threat Characteristic	**Probability of security breach**		
	Availability	***Integrity***	***Confident-iallity***
Politically explosive, high public interest	very often	occasionally	occasionally
High risk for (industry) espionage	very often	occasionally	very often
No specific characteristic	very often	occasionally	rare

TABLE II. PROTECTION NEEDS

Probability	**Value of Protection**			
	Low	***Normal***	***High***	***Very High***
Very rare	low	low	low	normal
Rare	low	normal	normal	high
Occasionally	low	normal	normal	high
Often	low	normal	high	high
Very often	low	high	high	very high

C. Selecting Requirements

The foundation for this step is a comprehensive list of non-functional requirements relevant to IT security or regarding legal constraints. Examples of such requirements are:

- In Germany, if the processed data is *personal*, then the contract with the cloud service provider must be compliant with the Federal Data Protection Act (Bundesdatenschutzgesetz).
- The computing center of the cloud provider should have a redundancy of N+1 for critical components in order to be able to offer high or very high availability.
- In order to ensure *very high* protection of integrity and confidentiality a strong authentication (i.e., two factor authentication) for users of the service is necessary.

 ISBN: 978-1-61208-388-9

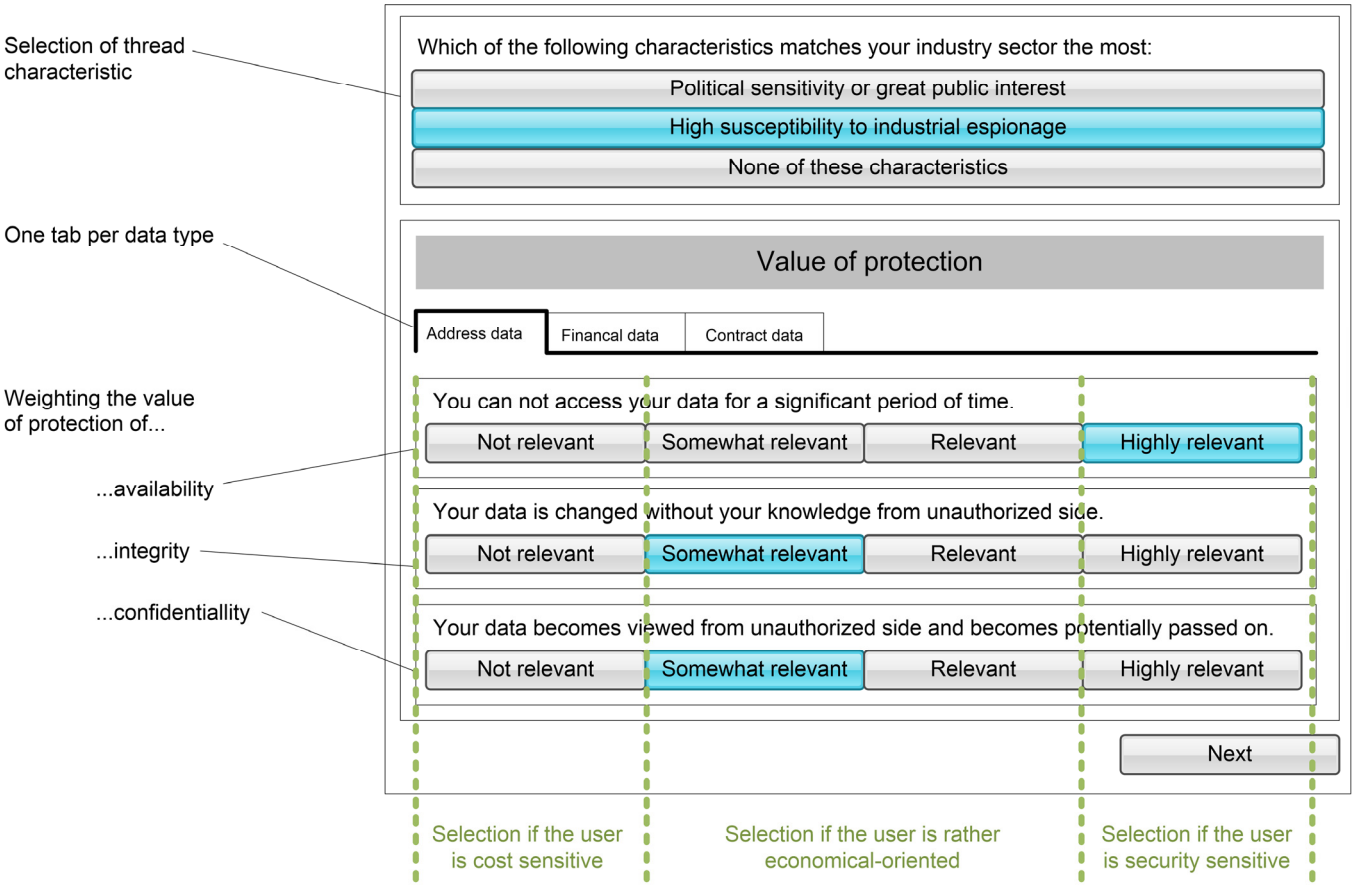


Figure 6. Screenshot of the input elements used to qualify the protection needs

A schema for conveniently expressing that list of requirements is depicted in Fig. 5.

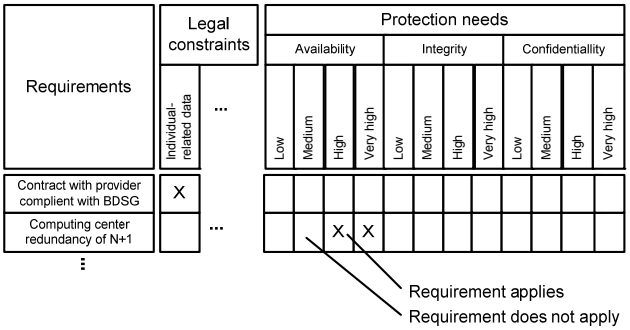


Figure 5. Schema for expressing the set of non-functional requirements

The objective of this last step is the selection of a subset of requirements from this list. This can be achieved by selecting the requirements which do apply when considering the given legal constraints as well as the individual protection needs.

VI. Application

The method described in the previous section was implemented as part of a prototype for a web-based search system for SMEs supporting the selection of adequate cloud services. The central components of this search system are:

- **Fn:** The first component of the system is responsible for the determination of functional requirements regarding a cloud services. In this step, the system supports the user by allowing a semantic extraction of functional terms from product websites specified by the user (i.e., the website of a product the user currently has in use and intends to replace by a cloud service). The technique for this semantic extraction of functional requirements is described in detail in [21]. The functional requirements are represented as a set, detailing for each identified function what kind of data (data types) this function processes and which legal constraints have to be obeyed in this context.
- **Non-Fn:** The second component implements the method described in this paper.
- **Search:** The last component performs a search in the service repository using the search profile defined by

 ISBN: 978-1-61208-388-9

the functional and non-functional requirements. For each service the fulfillment of the individual functional and non-functional requirements was stored in the repository. The search was performed by filtering out services which miss one or more of the identified requirements.

In the following, some background information regarding the implementation of the Non-Fn component will be given.

A. Preprocessing

Data types and legal constraints are automatically determined based on the functional requirements which were identified by the Fn component. In order to achieve this, the system uses a data set which allows a mapping from functional requirements to associated data types. Furthermore, the data set specifies if a data type has one or more legal constraints. The following legal constraints are supported by the system: *personal data, fiscal relevant data* and *security clearance*.

B. Qualifying Protection Needs

In order to gather the value of protection for each tuple *data type* / *protection target* as well as for selecting an adequate threat characteristic, corresponding questions and input elements are presented to the user (see Fig. 6). Then, the individual needs regarding the three protection targets (availability, integrity, confidentiality) are derived based on these inputs.

VII. EVALUATION

The first evaluation comprised the review of the presented method, as well as the list of non-functional requirements by two experts – one with expertise in IT security, one with expertise in both IT security and legal. Both experts did attest the list to successfully cover the necessary aspects of IT security as well as relevant legal aspects regarding the selection of cloud services. Based on the reviewers' feedback the method itself was further improved resulting in the form presented in Section V.

The second evaluation stage involved the assessment of the implemented prototype of the search system by users from five independent SMEs from the craft domain. The prototype was presented to each of these reviewers and afterwards an interview was conducted in order to get the reviewer's opinion regarding the search system and the underlying method. During the interviews, the reviewers attested the search system (together with the underlying method) to successfully assist users with the identification of relevant non-functional requirements. Hence, the system would reduce the amount of expertise required by users of SMEs as well as the associated effort of searching for appropriate cloud services.

VIII. CONCLUSION

In this paper, we have presented a method for identifying IT security and legal requirements regarding a cloud service. Also, details regarding the implementation of the method as a component of a service search as well as evaluation results were given. As the feedback of users of SMEs was quite promising we think that the underlying method is well suited for supporting the identification of cloud service requirements. Hence, we think that by utilizing the method in search systems for cloud services the process of finding appropriate cloud services can be simplified and accelerated as users do not have to identify these requirements on their own. Instead, a search system can handle all these individual requirements under the hood and just present the appropriate services to the user. If users are convinced that such a search system has taken into account all relevant requirements (in particular covering IT security and law) this could further increase the adoption of cloud computing by SMEs as users would have more trust that the selected service is appropriate for them.

Due to these promising results, we plan to further develop the service search system. In addition to improvements to the user experience, we intend to further fill the repository with various cloud services from different domains. This will help us to further asses the relevance of the presented method as well as the relevance of such search system as a whole.

ACKNOWLEDGMENT

This work was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference "01MD11041".

REFERENCES

[1] R. Sahandi, A. Alkhalil, and J. Opara-Martins, "Cloud computing from SMEs perspective: a survey-based investigation," Journal of Information Technology Management, vol. 24, No. 1, University of Baltimore, 2013, pp. 43-49.

[2] H. Kasper, H. Kett, and A. Weisbecker, Potenziale von Cloud Computing im Handwerk. Stuttgart: Fraunhofer Verlag, 2012.

[3] S. Lamberth and E. Hebisch, Sichere Cloud, Technical Report, Stuttgart: Fraunhofer IAO, 2011.

[4] H. Balzert, Lehrbuch der Softwaretechnik – Basiskonzepte und Requirements Engineering. 3rd ed., Heidelberg: Spektrum, 2009.

[5] T. Mather, S. Kumaraswamy, and S. Latif, Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance. O'Relly, 2009.

[6] R. L. Krutz and R. D. Vines, Cloud Security: A Comprehensive Guide to Secure Cloud Computing. Indianapolis: Wiley, 2010.

[7] Bundesamt für Sicherheit in der Informationstechnik (BSI): *Sicherheitsempfehlungen für Cloud Computing Anbieter*, Bonn, 2012.

[8] C. Lévy-Bencheton, L. Marinos, R. Mattioli, T. King, C. Dietzel, and J. Stumpf, Threat Landscape and Good Practice Guide for Internet Infrastructure, Report, European Union Agency for Network and Information Security (ENISA), 2015.

[9] R. Ko and S. Lee, Cloud Computing Vulnerability Incidents, Report, Cloud Security Alliance (CSA), 2012.

[10] Á. Geréd, A. Weiss, B. Becker, U. Huber, and C. Zeidler, Cloud Computing – Herausforderungen, Qualitätssicherung, Standards und Zertifizierung. Wien: EuroCloud.Austria, 2013.

 ISBN: 978-1-61208-388-9

[11] International Organization for Standardization (ISO): *ISO/IEC 27001:2013*.

[12] EuroCloud: *ECSA - EuroCloud Star Audit*. http://eurocloud-staraudit.eu/ [accessed: 10.07.2013].

[13] M. Bedner, Cloud Computing: Technik, Sicherheit und rechtliche Gestaltung. Dissertation, Kassel: University Press, 2013.

[14] J. Eckhardt, M. Hilber, R. Giebichenstein, F. Niemann, T. Helbing, and A. Weiss, Leitfaden Cloud Computing – Recht, Datenschutz und Compliance. Köln: EuroCloud Deutschland_eco, 2010.

[15] Bundesverband Informationswirtschaft, Telekommunikation und neue Medien (BITKOM): *Cloud Computing - Evolution in der Technik, Revolution im Business*, Berlin, 2009.

[16] M. Zhang, R. Ranjan, S. Nepal, M. Menzel, and A. Haller, "A Declarative Recommender System for Cloud Infrastructure Services Selection," 9th Int. Conf. on Economics of Grids, Clouds, Systems, and Services (GECON'12), 2012, pp. 102-113.

[17] S.-M. Han, M. M. Hassan, C.-W. Yoon, and E.-N. Huh, "Efficient Service Recommendation System for Cloud Computing Market," 2nd Int. Conf. on Interaction Sciences: Information Technology, Culture and Human, 2009, pp. 839-845.

[18] C. Perrin, The CIA triad, techrepublic.com, 2008. http://www.techrepublic.com/blog/it-security/the-cia-triad/ [accessed: 25.12.2014].

[19] C. Eckert, IT-Sicherheit, Konzepte – Verfahren – Protokolle, 8th ed., Munich: Oldenbourg, 2013.

[20] Bundesamt für Sicherheit in der Informationstechnik (BSI): *BSI-Standard 100-2*.

[21] A. Horch, C. Christmann, and H. Kett, "Automated Elicitation of Functional User Requirements for Supporting Cloud Service Search," Submitted to: 3rd Int. Conf. on Building and Exploring Web Based Environments (WEB 2015).

[22] V. Avelar, Guidelines for Specifying Data Center Criticality / Tier Levels, White Paper, Schneider Electric, 2011.

[23] Uptime Institute: *Data Center Site Infrastructure Tier Standard: Topology*, Santa Fe, 2010.

[24] J. Wollersheim, P. Hoberg, and H. Krcmar, Merkmale einer Servicebeschreibung für Cloud Services - V 0.9, 2013.

 ISBN: 978-1-61208-388-9

The UTOPIA Video Surveillance System Based on Cloud Computing

Jong Won Park, Chang Ho Yun, Hak Geun Lee, Chul Sang Yoon, Hae Sun Jung, Yong Woo Lee
School of Electrical & Computer Engineering
The Smart City (Ubiquitous City) Consortium, the University of Seoul
Seoul, South Korea
{comics77, touch011, withteas, csyoon2014, banyasun, ywlee}@uos.ac.kr

***Abstract*— Smart City is an intelligent city which satisfies human beings' desire to enjoy IT services with any device, anytime, anywhere and a future city model based on Internet of Everything (IoE) or Internet of Things (IoT). It includes a lot of video cameras which are networked together and the networked video cameras enable a lot of smart city services. The networked video cameras generate huge amount of video information, real big data for the smart city all the time and the smart city should process the big data in real-time in most cases. It requires a lot of computational power and usually takes a lot of time thus it is a very challenging task. Cloud computing can be a good solution to address this matter and there are many cloud computing methodologies. This paper presents our own methodology to analyze the video images using MapReduce. The methodology was implemented in our smart city middleware called SOUL, which was designed for our smart city called UTOPIA. Some of the implemented system is introduced in this paper. This paper also introduces smart analyzing functions of the video surveillance system in SOUL and how they use the scalable video streaming of SOUL to acquire the video data from the networked surveillance video cameras. The system is easy to be deployed, flexible and reliable. The performance evaluation was experimented and we found that our proposed system worked well. Some analyzed results of the performance evaluation are presented in this paper.**

Keywords-Smart City; Cloud Computing; Smart Video Surveillance System; MapReduce; Networked Video System; Image Processing; Big Data.

I. INTRODUCTION

A smart city is an ICT based city, converges every possible information system, such as residential, environmental, medical, business, governmental, social information systems and the whole activities in a city into a virtual system or a global system which works for human beings. It has key aspects such as smart infrastructure, smart citizen care and smart administration which include smart traffic management, smart ecological environment management, smart energy management, etc.

In the smart city, there are a lot of networked video cameras to support smart city services and they generate huge data continuously. It is usually required that the smart city should process the huge and continuous video data, real big data in real time [1][2]. We think that this requirement can be successfully solved by cloud computing technology since cloud computing can provide the computing resources to process big data successfully.

More specifically speaking, Hadoop [3] can be a useful solution among many open source solutions for cloud computing, since Hadoop stores the data in Hadoop Distributed File System (HDFS) and provides the platform that can process them with simple MapReduce programming model [4]. It enables us to process the big semi-structured data and big unstructured data that were hard to be processed before. Indeed, the video data which continuously produced in smart city are an unstructured data and big data.

This paper presents a research result about the smart video surveillance system based on cloud computing for smart city. The smart video surveillance system collects big video data through scalable video streaming which smoothly processes big data traffic from large number of networked video cameras even with limited bandwidth and process the big data with MapReduce model.

The contributions of this paper can be described as follows:

1. This paper presents a video surveillance system for smart city which process big video image data with scalable video streaming and cloud computing in real time efficiently. Thus, it contributes to big data processing, video image data processing, cloud computing, scalable video streaming and real time processing and the video surveillance system
2. This paper proposes the video surveillance system was implemented in in our smart city middleware called SOUL, which was designed for our smart city called UTOPIA. Thus, it contributes to the field of smart city and the field of the middleware system for the smart city.
3. This paper presents the result of performance evaluation.

This paper is organized as follows. Section 2 gives overview of the SOUL smart video surveillance system in UTOPIA. Section 3 explains the architecture and operational principle of video surveillance system in SOUL. Section 4 explains the details of the cloud computing based big video data processing. Section 5 explains the performance evaluation work. Section 6 explains related works and compares them with our work. Finally, Section 7 gives the conclusion.

 ISBN: 978-1-61208-388-9

II. SOUL SMART VIDEO SURVEILLANCE SYSTEM IN UTOPIA

In smart city, citizens can enjoy various integrated smart city services. In order to support the smart city services, we designed the ICT based smart city model which has 3-tier architecture and we call UTOPIA. UTOPIA consists of the 3 tiers: the Body tier, the Processing tier and the Presentation & Remote Control tier.

The Body tier in UTOPIA is like the body of the human being. This tier includes various kinds of information collection devices such as networked sensors, networked video cameras, microphones, GPSs and etc. It collects the various kinds of data in a smart city through the converged network.

The Processing tier works as the brain of smart city. It smartly processes the data from the Body tier. In smart city, critical decisions should be made in timely manners for various integrated smart city services. For it, we invented a smart middleware system which we call SOUL. SOUL has multi-layered architecture for many benefits such as expandability, decreasing complexity, component reusability, etc. It is the heart of the Processing tier.

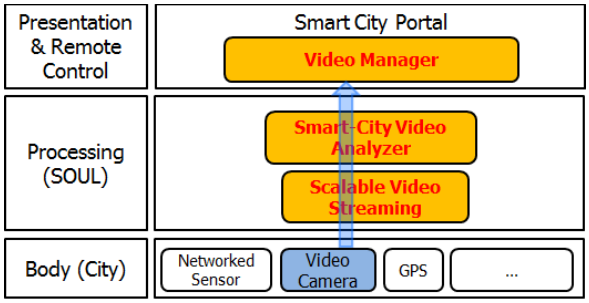


Figure 1. The architecture of UTOPIA and SOUL.

SOUL has the common device interface to various kinds of information collection devices such as networked sensors, networked video cameras, GPSs and etc. It manages the converged network, collects data from the Body tier through the scalable video streaming function, and sends the collected data to the smart city video analyzer. It supports the variable kinds of protocols and a general gateway like function for the Body tier [1][2][5].

SOUL processes the acquired data and makes smart services using the ontology-based context data management. It has intelligent inference engine and does automatic service discovery, service deployment and service execution based on inferred results so that it can provide context-aware smart services [6]. For it, SOUL manages computer resource, does cloud computing and Grid computing [7]. SOUL has the interface to the Presentation and Remote Control tier.

The Presentation & Remote Control tier provides the interface to manage the smart city and provides various smart city services to end users. The smart city portal is an example. Through it, the administrator can control and monitor the smart city. Also, the citizens enjoy the smart city services.

The portal provides the cloud computing interface to end users [8][9][10][11][12] and the tele-management interface to control remote devices such as water pumps, fire doors, emergency devices, remote networked video cameras, etc. [5][13][14][15].

The content in this paper includes our works such as the network adaptive scalable video streaming, water quality control and fire accident management, which we do not deal in detail in this paper [16]. The network adaptive scalable video streaming with the scalable video coding techniques is used in order to save network resources [17][18][19]. Figure 1 shows the concept of the SOUL Smart Video Surveillance System in UTOPIA.

III. VIDEO SURVEILLANCE SYSTEM IN SOUL

The architecture of the SOUL Smart City Video Surveillance System is shown in Figure 2. The SOUL Smart City Video Surveillance System consists of Video Manager which belongs to Presentation and Remote Control tier, Smart-City Video Analyzer and Scalable Video Streaming which belongs to SOUL as shown in Figure 1.

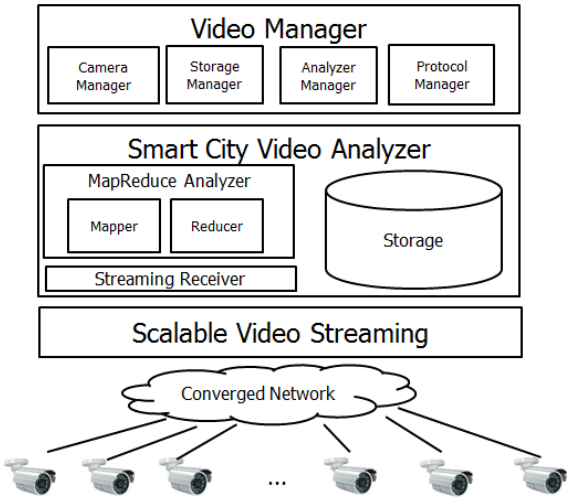


Figure 2. The architecture of SOUL Video Surveillance System.

Video Manager has several modules to manage Smart City Video Analyzer and Scalable Video Streaming. Camera Manager monitors and manages the status of cameras linked to smart city. Storage Manager monitors the status of Storage in Smart City Video Analyzer. Analyzer Manager changes the resolution, codec, and frame rates of video from user's request and monitors the status of MapReduce Analyzer. Protocol Manager manages video streaming protocols such as Real Time Streaming Protocol (RTSP) and Real Time Messaging Protocol (RTMP). Smart City Video Analyzer consists of Streaming Receiver, MapReduce Analyzer and Storage. The Streaming Receiver collects input streams from Scalable Video Streaming. It also delivers the input stream from outside SOUL to inside of SOUL. HDFS is used to store the big video data from Streaming Receiver.

MapReduce Analyzer will be described in the details in the next section.

Smart city need a lot of video cameras in order to manage smart city efficiently. Thus, SOUL was designed to efficiently manage the large amount of video data and has the scalable video streaming component to smoothly manage the video data traffic from large number of networked video cameras. It uses a noble mechanism which controls admissions of clients, extracts bit-streams, and allocates appropriate channels to do the context-aware and the network-adaptive video streaming as shown in Figure 3. The noble mechanism can save system resources and network resources through self-learning using scalable video coding techniques [19].

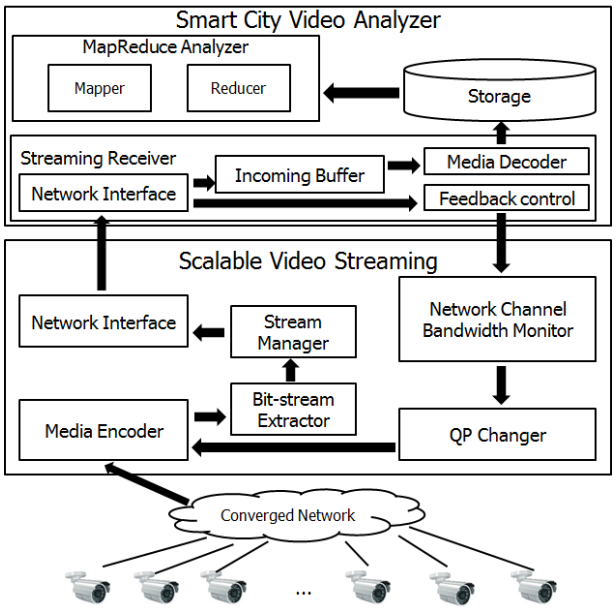


Figure 3. The operational principle of of SOUL Video Surveillance System.

The operation of the scalable video streaming is as follows. Media encoder encodes the video data acquired from each camera linked to the converged network in the smart city. When scalable video streaming encodes video data, it checks the changing instructions from the QP (quantization parameter) changer. Thus, the acquired video data are encoded with various QP values, determined by the QP changer with information from the network channel bandwidth monitor. The network channel bandwidth monitor acts as a network analyzer. It analyzes network bandwidth in real time using a feedback signal and the strength of radio wave signals and the available bandwidth of stream receiver.

In the initialized state, the Scalable Video Streaming sends a message to Streaming Receiver of Smart City Video Analyzer, which responds with their available bandwidth. This information is sent from the feedback control. If the signal is strong, Streaming Receiver considers the good condition of network channel. In contrast, if the signal is weak, the network channel is considered to be in a bad condition.

The feedback control module of Streaming Receiver helps the network channel bandwidth monitor to evaluate the condition of the network channel. After receiving video data, Streaming Receiver responds.

If the channel conditions change very frequently, the feedback control sends information in a minimum state until the channels stabilize. At the very least, this information is used again to evaluate the user network. Thus, feedback control module sends only the information of current bandwidth of Streaming Receiver to scalable video streaming. Scalable video streaming determines the QP value using these data.

The network channel bandwidth monitor observes the network channel condition continuously. This module evaluates the available bandwidth and communicates with the QP changer module to control the encoding rate of the encoder based on a predetermined QP value. Then, The QP changer module regulates the encoding level.

The live video data are encoded by the determined QP value in the media encoder, which is in the network adaptive live streaming module. Encoded video data are transmitted through a bit-stream extractor and a stream manager, which packetize data appositely. These scalable video data packets are separated to adapt to various network conditions.

IV. CLOUD COMPUTING BASED BIG VIDEO DATA PROCESSING

The operation of “MapReduce Analyzer” is shown in Figure 4. Video input data are split by static offset. The key of map phase input is the path of video file as shown in table 1. The value of map phase is each offset. Mapper trims the video file according to path of video file and offset. And mapper analyzes trimmed video through the “Analyzing Module”. After that, output of the mapper has the name of video as the key and analyzed data as the value.

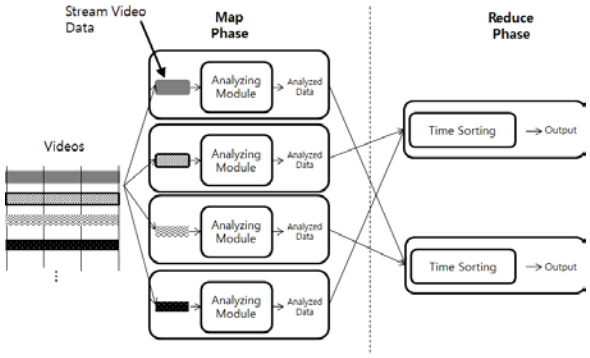


Figure 4. The operation of MapReduce Analyzer.

After generating output of the mapper, intermediate data is sorted according to the name of video. The Reducer receives the pair value of <video ID, list(analyzed data)> and

sorts the analyzed data according to the time and, finally, writes the output to HDFS.

TABLE I. THE KEY VALUE PAIR OF MAPREDUCE

Map Phase	Input	<video path, offset>
	Output	<video ID, analyzed data>
Reduce Phase	Input	<video ID, list(analyzed data)>

Splitting the video source is important in our approach. Y.S. Wu et. al. [20] splits the raw video by chunk size to encode but we do not and use compressed video data. The group of picture (GOP) in H.264 contains two frame of picture types such as the Key frame and the non-Key frames. Key frame (I-frame) is the reference frame which represents a fixed image and is independent of other frame type. In our approach, we split the video static GOP length to avoid to loss the key frame and assume that the processing time of computer vision algorithm is related with the frame length.

V. PERFORMANCE (EVALUATION)

There are two main approaches to object detection: temporal difference and background subtraction [21]. We use temporal difference to evaluate performance. The application was implemented in Apache Hadoop MapReduce environment 0.23.0 and was written in Java language. The "x264 codec" which is one of open sources for H.264 codec was used. FFmpeg's libavutil and ffprobe libarary were used to handle video image. Analyzing Module uses OpenCV 2.4.3 that is a very popular computer vision library and is written in C++ language. These libraries were installed on every cluster node, due to dependencies problems between the libraries.

We used a 14 nodes cluster, where each of the 14 nodes was identical Intel core i5 760 2.8 Ghz Processors and had 8 GB of physical memory. There, all nodes were connected to a Giga-bit Ethernet switch and ran a Ubuntu 12.04 LTS 64bit server edition. The JVM version 1.6.0_22 was used. Also, all nodes used mounted HDFS through "hdfs-fuse". The number of Mappers was two per a node.

Input video data had the resolution 1920x1080 (FHD), total frame length was 36000 frames, average of bitrate was 19.4Mbit/s, the length of GOP was 25, the time of video was about 10 minute and the size of video data was 1.35GByte. When two frames were processed, it took 0.355 seconds. When 36000 frames were processed, it took about 21 minute 18 seconds.

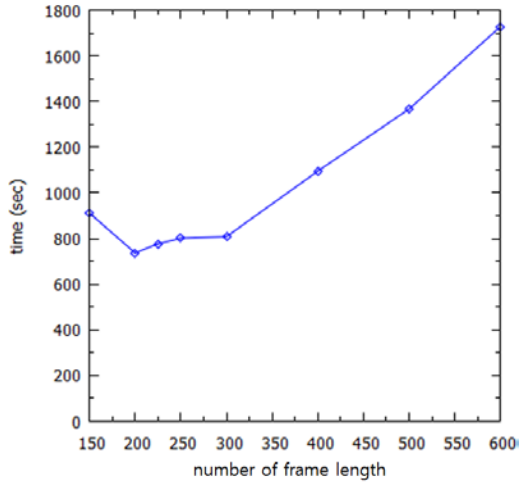


Figure 5. The processing time according to the number of frame length.

Figure 5 shows the results when the frame split size was varied with the fixed input that consisted of 2 videos which had 36000 length of GOP each. It was seen that when the splitting size was smaller than 200, the processing time linearly decreased and when the splitting size was smaller than 200, the processing time was increased.

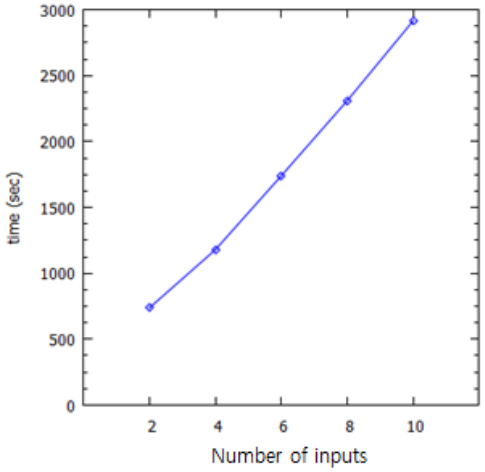


Figure 6. The Processing time according to number of input when the frame split size was 200.

Figure 6 shows the processing time when the frame split size was fixed at 200 and the number of input was varied. There, we see that the processing time increases linearly when the number of input is increased.

VI. RELATED WORK

Video surveillance system usually has the problem of big data generated from video cameras. Traditional distributed

system technologies had been unable to solve the big data problem in real time [21][22][23]. The problem seems to be manageable with cloud computing and video surveillance systems seems to have new era.

C.F. Lin et al. [24] proposed a cloud based video recorder system. They propose it to replace traditional video recorder such as the network video recorder (NVR) and the digital video recorder (DVR). They use HDFS as scalable and backupable storage in their distributed replication mechanism and deal with deployment design for public cloud and private/hybrid cloud.

D.A. Rodriguez-Silva et al. [25] suggest traditional video surveillance architecture combined to cloud computing. The system consists of three parts such as the web server, the cloud processing servers and the cloud storage servers. It uses Amazon S3 storage to store video and the SSL protocol. They present a fault-tolerance mechanism which works between processing servers and storage servers.

Y.S. Wu et al. [20] present an architecture to solve the network bottleneck which the centralized video data processing system usually has. It has a well-developed peer to peer (P2P) architecture and HDFS. It proposes its own scheduling algorithm.

M.S. Hossain et al. [26] present a research result regarding virtual machine (VM) resource allocation for cloud-based video surveillance platform. The platform is a general framework of video surveillance service composition in cloud. It deals with linear programming formulation with migration control and simulation using Amazon Web Service (AWS).

R. Pereira et al. [27] propose the Split&Merge architecture to reduce video encoding times, regardless of the video data size. B. White et al. [28] present implementation research of their algorithms in several different kinds of computer systems.

VII. Conclusion and Future Work

This paper presented a cloud-computing based smart video surveillance system for smart city which uses MapReduce to process a huge amount of video data in real time with the performance evaluation. Here, we used 1920x1080(FHD) resolution video data and HDFS as storage to store video. We analyzed the processing time according to the number of frame per mapper. Tracing the optimal splitting size of input data and the processing time according to the number of nodes showed the linearity in the system performance. We believe our system is very successful in managing smart city. For example, it was very useful in managing urban traffic, fire accident, etc. In our future work, we hope to experiment when the resolution and the other factors of video data was varied with various kinds of distributed storage solutions such as GlusterFS, Lustre and Ceph. We also want to optimize our system, as well as to add various kinds of new functions.

Acknowledgment

This work was supported by the 2014 Research Fund of the University of Seoul (Yong Woo Lee: the corresponding author). Patents in Korea and in the United State of America are registered and are in pending for the contents of this paper.

References

[1] H. S. Jung, C. S. Jeong, Y. W. Lee, and P. D. Hong, "An Intelligent Ubiquitous Middleware for U-City: SmartUM," Journal of Information Science and Engineering, vol. 25, Issue 2, Mar. 2009, pp. 375-388.

[2] H. S. Jung, J. K. Baek, C. S. Jeong, Y. W. Lee, and P. D. Hong, "Unified Ubiquitous Middleware for U-City," Proc. International Conference on Convergence Information Technology 2007 (ICCIT 07), Nov. 2007, pp. 2347-2379.

[3] Apache Hadoop, [Online], Nov. 2014, Available from: http://hadoop.apache.org/

[4] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," Proc. 6th Symposium on Operating Systems Design and Implementation (OSDI), Dec. 6-8 2004, pp.137-150.

[5] C. H. Yun, H. Han, H. S. Jung, H. Y. Yeom, and Y. W. Lee, "Intelligent Management of Remote Facilities through a Ubiquitous Cloud Middleware," Proc. IEEE International Conference on Cloud Computing (CLOUD 09), Sep. 2009, pp. 65-71.

[6] C. H. Yun, Y. W. Lee, and H. S. Jung, "An Evaluation of Semantic Service Discovery of a U-City Middleware," Proc. 12th International Conference on Advanced Communication Technology (ICACT 10), Feb. 2010, pp. 600-603.

[7] J. W. Park, C. H. Yun, S. G Gyu, H. Y. Yeom, and Y. W. Lee, "Cloud Computing Platform for GIS Image Processing in U-City," Proc. 13th International Conference on Advanced Communication Technology (ICACT 11), Feb. 2011, pp. 1151-1155.

[8] Y. W. Lee and S. W. Rho, "U-City Portal for Smart Ubiquitous Middleware," Proc. 12th International Conference Advanced Communication Technology (ICACT 10), vol. 1, Feb. 2010, pp. 609-613.

[9] S. W. Rho, C. H. Yun, and Y. W. Lee, "Provision of U-city Web Services Using Cloud Computing," Proc. 13th International Conference on Advanced Communication Technology (ICACT 11), Feb. 2011, pp. 1545-1549.

[10] P. D. Hong and Y. W. Lee, "A Grid Portal for Grid Resource Information Service," Proc. 13th International Conference on Advanced Communication Technology (ICACT 11), Feb. 13-16 2011, pp. 597-602.

[11] J. W. Park, C. H. Yun, H. S. Jung, and Y. W. Lee, "Mobile Cloud and Grid Web Service in a Smart City," Proc. The Fifth International Conference on Cloud Computing, GRIDs, and Virtualization, May 25-29 2014, pp. 20-25.

[12] J. W. Park, J. O. Kim, J. W. Park, C. H. Yun, and Y. W. Lee, "Mobile Cloud Web-Service for U-City," Proc. International Conference on Cloud and Green Computing 2011, Dec. 2011, pp.1161-1065.

[13] J. W. Park, C. H. Yun, H. S. Jung, and Y. W. LEE, "Visualization of Urban Air Pollution with Cloud Computing," Proc. IEEE World Congress on Services (SERVICES) 2011, Jul. 4-9 2011, pp. 578-583.

[14] H. K. Park, J. W. Park, C. H. Yun, and Y. W. Lee, "Distributed and parallel processing of noise information for 3D GIS in a U-city," Proc. 13th International Conference on Advanced Communication Technology (ICACT) 2011, Feb. 13-16 2011, pp. 855-858.

[15] J. W. Park et al., "Cloud Computing for Online Visualization of GIS Applications in Ubiquitous City," Proc. Cloud Computing 2010, Nov. 21-26 2010, pp. 170-175.

 ISBN: 978-1-61208-388-9

[16] Y. W. Lee, H. W. Kim, H. Y. Yeom, C. H. Yun, S. W. Rho, and H. S. Jung, "Water Quality Control using E-science Technology," Proc. Fifth IEEE International Conference on e-science (e-Science 09), Dec. 2009.

[17] S. W. Ahn, H. S. Jung, Y. W. Lee, and C. Yoo, "Network Condition Adaptive Real-time Streaming of an Intelligent Ubiquitous Middleware for U-City," Proc. 4th International Conference on Ubiquitous Information Technologies & Applications (ICUT 09), Dec. 2009, pp. 1-5.

[18] E. S. Ryu, H. S. Jung, Y. W. Lee, H. Yoo, C. S. Jeong, and S. W. Ahn, "Content-aware and Network-adaptive Video Streaming of a Ubiquitous Middleware," Proc. 11th International Conference on Advanced Communication Technology (ICACT 09), Feb. 2009, pp. 1458-1461.

[19] S. W. Ahn, H. S. Jung, C. Yoo, and Y. W. Lee, “NARS: Network Bandwidth Adaptive Scalable Real-Time Streaming for Smart Ubiquitous Middleware," Journal of Internet Technology, Vol.14, No.2, Mar. 2013, pp. 217-230.

[20] Y. S. Wu, Y. S. Chang, T. Y. Jang, and J. S. Yen, “An Architecure for Video Surveillance Service based on P2P and Cloud Computing," Proc. Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing(UIC/ATC), Sep. 2012, pp. 661-666.

[21] M. Valera and S. A. Velastin, "Intelligent distributed surveillance systems: a review," Proc. IEE Vision, Image and Signal Processing, vol. 152, no. 2, Apr. 2005, pp. 192-204.

[22] H. Detmold, A. V. D. Hengel, A. R. Dick, K. E. Falkner, D. S. Munro, and R. Morrison, "Middleware for Distributed Video Surveillance," IEEE Distributed systems Online, vol. 9, no. 2, Feb. 2008, pp. 1-10.

[23] A. Hampaur et al., "S3 : The IBM Smart Surveillance System: From Transactional Systems to Observational Systems," Proc. IEEE International Conference on Acoustics, Speech and Signal Processing 2007 (ICASSP 07), Apr. 2007, pp. 1385-1388.

[24] C. F. Lin, S. Yuan, M. Leu, and C. Tsai, "A Framework for Scalable Cloud Video Recorder System in Surveillance Environment," Proc. Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC), Sep. 2012, pp. 655-660.

[25] D. A. Rodriguez-Silva, L. Adkinson-Orellana, F.J. Gonz’lez-Castaño, I. Armino-Franco, and D. Gonz’lez-Martinez, "Video Surveillance Based on Cloud Storage," Proc. IEEE Cloud Computing(CLOUD), Jun. 2012, pp. 991-992.

[26] M. S. Hossain, M. M. Hassan, M. A. Qurishi, and A. Alghamdi, "Resource Allocation for Service Composition in Cloud-based Video Surveillance Platform," Proc. IEEE International Conference on Multimedia and Expo Workshops(ICMEW), Jul. 2012, pp. 408-412.

[27] R. Pereira, M. Azambuja, K. Breitman, and M. Endler, "An Architecture for Distributed High Performance Video Processing in the Cloud," Proc. IEEE 3rd International Conference on Cloud Computing, Jul. 2010, pp. 482-489.

[28] B. White, T. Yeh, J. Lin, and L. Daivs, "Web-Scale Computer Vision using MapReduce for Multimedia Data Mining," Proc. the 10th International Workshop on Multimedia Data Mining(MDMKDD), Jul. 2010, pp. 1-10.

 ISBN: 978-1-61208-388-9

DesktopCloudSim: Simulation of Node Failures in the Cloud

Abdulelah Alwabel, Robert Walters, Gary Wills
School of Electronics and Computer Science
University of Southampton
Southampton, UK
e-mail: {aa1a10, rjw1, gbw}@ecs.soton.ac.uk

***Abstract*—Simulation tools are commonly used by researchers to simulate Clouds in order to study various research issues and test proposed solutions. CloudSim is widely employed to simulate Cloud computing by both academia and industry. However, it lacks the ability to simulate failure events which may occur to physical nodes in the infrastructure level of a Cloud. This paper proposes DesktopCloudSim tool as an extension developed to overcome this shortage. In order to demonstrate the effectiveness of this tool, we evaluate the throughput of simulating a private Cloud built on top of faulty nodes based on empirical data collected from NotreDame Desktop Grid.**

Keywords-Cloud; CloudSim; DesktopCloudSim; Failure; Nodes.

I. INTRODUCTION

Cloud computing has emerged with a promise to improve performance and reduce running costs. The services of Cloud computing are provided by Cloud Service Providers (CSPs). Traditionally, CSPs use a huge number of computing resources in the infrastructure level located in datacentres. Such resources are claimed to have a high level of reliability which makes them resilient to failure events [1]. However, a new direction of Cloud has recently emerged with an aim to exploit normal Desktop computers, laptops, etc. to provide Cloud services [2]. This kind of Cloud can be called Desktop Clouds [3]. In contrast to the traditional way of CSP which uses a huge number of computing resources that are dedicated to be part of the Cloud. Throughout this paper, the term Traditional Cloud refers to this traditional way of Clouds.

The cost-effectiveness of Desktop Clouds is the main advantage over Traditional Clouds. Researchers in Desktop Clouds can benefit from Cloud services at little cost, if not free. However, such feature comes with a price. The nodes of a Desktop Cloud are quite volatile and prone to failure without prior knowledge. This may affect the throughput of tasks and violate the service level agreement. The throughput is defined as the number of successful tasks submitted to be processed by Virtual Machines (VMs). Various VM allocation mechanisms can yield different variations of throughput level in the presence of node failures.

VM allocation mechanism is the process of allocation requested VMs by Cloud's users to physical machines (PMs) in the infrastructure level of a Cloud. The contribution of this paper can be summarised into: (i) it proposes and describes the DesktopCloudSim as being an extension for CloudSim simulation toolkit; (ii) it investigates the impact of failure events on throughput and (iii) three VM mechanisms: FCFS, Greedy and RoundRobin mechanisms are evaluated in terms of throughput using DesktopCloudSim. The reminder of this paper is organised as follows: Section II discusses Desktop Cloud as being a new direction of Cloud computing. Section III proposes the simulation tool that extends CloudSim. The section starts by reviewing CloudSim to show the need to extend it. The section, then, reviews some VM allocation mechanisms. Next section demonstrates experiments conducted to evaluate the impact of node failures in a Desktop Cloud based on empirical data of failures in NotreDame nodes. The results are then analysed and discussed in Section V. Several related works are reviewed in Section VI. Finally, a conclusion and future work insights are given in the last section.

II. DESKTOP CLOUD

The success of Desktop Grids motivates the idea of harnessing idle resources to build Desktop Clouds. Hence, the term Desktop comes from Desktop Grids because both of Desktop Clouds and Desktop Grids are based on Desktop PCs and laptops etc. Similarly, the term Cloud comes from Cloud as Desktop Cloud aims to provide services based on the Cloud business model. Several synonyms for Desktop Cloud have been used, such as Ad-hoc Cloud [4], Volunteer Cloud [2], Community Cloud [5] and Non-Dedicated Cloud [6]. The literature shows that very little work has been undertaken in this direction.

There are some differences between Desktop Clouds and Traditional Clouds. Firstly, the infrastructure of Desktop Cloud consists of resources that are non-dedicated, i.e., not made to be part of Cloud infrastructure. Desktop Cloud helps in saving energy since it utilises already-running undedicated resources which would otherwise remain idle. Some studies show that the average percentage of local resources being idle within an organisation is about 80% [8]. It is shown that an idle machine can consume up to 70% of the total power

 ISBN: 978-1-61208-388-9

consumed when it is fully utilized according to [9]. On the contrary, the infrastructure of Traditional Clouds is made of a large number of dedicated computing resources. Traditional Clouds have a negative impact on the environment since their data centres consume massive amounts of electricity for cooling these resources.

Secondly, resources of Desktop Clouds are quite distributed across the globe, whereas they are limited in Traditional Cloud to several locations in data centres. Furthermore, nodes in Desktop Cloud are highly volatile due to the fact that they can be down unexpectedly without prior notice. Node failures can happen for various reasons such as connectivity issues, machine crashing or simply the machine becomes busy with other work by its owner takes priority. High volatility in resources has negative impact on availability and performance [7]. Although, resources in both Traditional Cloud and Desktop Cloud are heterogeneous, they are even more heterogeneous and dispersed in Desktop Cloud. Traditional Clouds are centralised, which leads to the potential that there could be a single point of failure issue if a Cloud service provider goes out of the business. In contrast, Desktop Clouds manage and offer services in a decentralised manner. Virtualisation plays a key role in both Desktop Clouds and Traditional Clouds.

Desktop Clouds can be confused with similar distributed systems, specifically Desktop Grids. Both Desktop Clouds and Desktop Grids share the same goal that is exploiting computing resources when they become idle. The resources in both systems can be owned by an organisation or denoted by the public over the Internet. Both Desktop Grids and Desktop Clouds can use similar resources. Resources are volatile and prone to failure without prior knowledge. However, Desktop Grids differ from Desktop Clouds in the service and virtualisation layers. Services, in Desktop Clouds, are offered to clients in an elastic way. Elasticity means that users can require more computing resources in short term [10]. In contrast, the business model in Desktop Grids is based on a 'project oriented' basis which means that every user is allocated a certain time to use a particular service [11]. In addition, Desktop Grids' users are expected to be familiar with details about the middleware used in order to be able to harness the offered services [12]. Specific software needs to be installed to computing machines in order to join a Desktop Grid. Clients in Desktop Clouds are expected to have little knowledge to enable them to just use Cloud services under the principle ease of use. Desktop Grids do not employ virtualisation to isolate users from the actual machines while virtualisation is highly employed in Desktop Clouds to isolate clients from the actual physical machines.

III. DESKTOPCLOUDSIM

DesktopCloudSim is an extension tool proposed to simulate failure events happening in the infrastructure level based on CloudSim simulation tool. Therefore, this section starts by a brief discussion of CloudSim. The extension tool, DesktopCloudSim, is presented next. DesktopCloudSim is used to evaluate VM allocation mechanisms, thus the last subsection in this section discusses traditional mechanisms that are used by open Cloud middleware platforms.

A. CLOUDSIM

CloudSim is a Java-based discrete event simulation toolkit designed to simulate Traditional Clouds [13]. A discrete system is a system whose state variables change over time at discrete points, each of them is called an event. The tool was developed by a leading research group in Grid and Cloud computing called CLOUDS Laboratory at The University of Melbourne in Australia. The simulation tool is based on both GridSim [14] and SimJava [15] simulation tools.

CloudSim is claimed to be more effective in simulating Clouds compared to SimGrid [16] and GroudSim [17] because CloudSim allows segregation of multi-layer service (Infrastructure as a Service, Platform as a Service and Software as a Service) abstraction [13]. This is an important feature of CloudSim that most Grid simulation tools do not support. Researchers can study each abstraction layer individually without affecting other layers.

CloudSim can be used for various goals [18]. First, it can be used to investigate the effects of algorithms of provisioning and migration of VMs on power consumption and performance. Secondly, it can be used to test VM mechanisms that aim at allocating VMs to PMs to improve performance of VMs. It is, also, possible to investigate several ways to minimise the running costs for CSPs without violating the service-level agreements. Furthermore, CloudSim enables researchers to evaluate various scheduling mechanisms of tasks submitted to running VMs from the perspective of Cloud brokers. Scheduling mechanism can help in decreasing response time and thus improve performance.

Although CloudSim is considered the most mature Cloud simulation tool, the tool falls short in providing several important features. The first is that does not simulate performance variations of simulated VMs when they process tasks [18]. Secondly, service failures are not simulated in CloudSim [19]. The service failures include failures in tasks during running time and complex overhead of complicated tasks. Furthermore, CloudSim lacks the ability to simulate dynamic interaction of nodes in the infrastructure level. CloudSim allows static configuration of nodes which remain without change during run time. Lastly, node failures are not included in CloudSim tool. DesktopCloudSim enables the simulation of dynamic nodes and node failures while performance variations and service failures are simulated by other tools. Section VI discusses those tools.

B. The Architecture of DesktopCloudSim

Simulation is necessary to investigate issues and evaluate solutions in Desktop Clouds because there is no real Desktop Cloud system available on which to run experiments. In addition, simulation enables control of the configuration of the model to study each evaluation metric. In this research, CloudSim is extended to simulate the resource management model. CloudSim allows altering the capabilities of each host machines located in the *data centre* entity in the simulation

 ISBN: 978-1-61208-388-9

tool. This feature is very useful for experimentations, as it is needed to set the infrastructure (i.e., physical hosts) to have an unreliable nature. This can be achieved by extending the *Cloud Resources* layer in the simulation tool. Figure 1. Depicts the layered architecture of CloudSim combined with an abstract of the DesktopCloudSim extension.

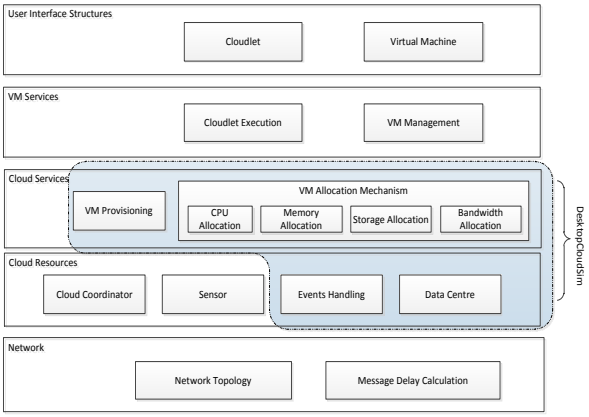


Figure 1. DesktopCloudSim Abstract

Figure 2. shows the components of DesktopCloudSim. The simulation starts by reading failure trace file(s). The trace files contain the specifications of the simulated nodes and failure events. The *Failure Analyser* component analyses the files of failures to send node specifications to *Create Nodes* component and failure events to *Failure Injection* component. Node specifications are the physical specifications of nodes, such as CPU, RAM. etc. *Create Nodes* component creates the nodes of a Desktop Cloud according to the given specification. *Failure Injection* component receives failure events from the *Failure* Analyser to inject failures into associated nodes during run time. The *Failure Injection* component informs the *VM Mechanism* unit if a node is failed to let it restart failed VMs on another alive node or nodes. *VM Provisioning* component provisions VMs to clients to execute tasks.

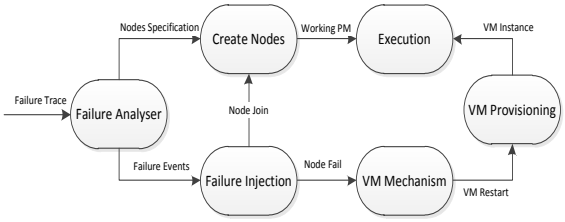


Figure 2. DesktopCloudSim Model

C. VM Allocation Mechanisms

Several VM allocation mechanisms that are employed in open Cloud platforms are discussed in this subsection. VM allocation mechanisms are: (i) Greedy mechanism which allocates as many VMs as possible to the same PM in order to improve utilisation of resources; (ii) RoundRobin mechanism allocates the same number of VMs equally to each PM; and (iii) First Come First Serve (FCFS) mechanism allocates a requested VM to the first available PM that can accommodate it. These mechanisms are implemented in open source Cloud management platforms, such as Eucalyptus [20], OpenNebula [21] and Nimbus [22].

When a VM is requested to be instantiated and hosted to a PM, the FCFS mechanism chooses a PM with the least used resources (CPU and RAM) to host the new VM. The Greedy mechanism allocates a VM to the PM with the least number of running VMs. If the chosen PM cannot accommodate the new VM, then the next least VM running PM will be allocated. RoundRobin is an allocation mechanism, which allocates a set of VMs to each available physical host in a circular order without any priority. For example, suppose three VMs are assigned to two PMs. The RoundRobin policy will allocate VM1 to PM1 then VM2 to PM2 then allocate VM3 to PM1 again. Although these mechanisms are simple and easy for implementation, they have been criticised for being underutilisation mechanisms, which waste energy [23]. The FCFS mechanism is expected to yield the lowest throughput among the aforementioned mechanisms because it assigns VMs to PMs in somehow random manner.

IV. EXPERIMENT

The experiment is conducted to evaluate VM mechanisms mentioned in Section III.C. There are two input types needed to conduct the experiment. The first input is the trace file that contains failure events happening during the run time. Failure trace files are collected from an online archive. Subsection IV.A discusses further this archive. The second input set is the workload submitted to the Desktop Cloud during running time. Subsection IV.B talks about this workload.

A. Failure Trace Archive

Failure Trace Archive (FTA) is a public repository containing traces of several distributed and parallel systems [24]. The archive includes a pool of traces for various distributed systems including Grid computing, Desktop Grid, peer-to-peer (P2P) and High Performance Computing (HPC). The archive contains timestamp events that are recorded regularly for each node in the targeted system. Each event has a state element that refers to the state of the associated node. For example, an event state can be unavailable which means this node is down at the timestamp of the event. The unavailable state is considered a failure event throughout this report. The failure of a node in an FTA does not necessarily mean that this node is down. For example, a node in a Desktop Grid system can be become unavailable because its owner decides to leave the system at this time.

The Notre Dame FTA is collected from the University of Notre Dame. The trace represents an archive of a pool of heterogeneous resources that have run for 6 months within the University of Notre Dame during 2007 [25]. Each month is provided separately representing the behaviour of nodes located in the University of Notre Dame. The FTA contains 432 nodes for month 1, 479 nodes for month 2, 503 nodes for month 3, 473 nodes for month 4, 522 nodes for month 5 and 601 nodes for month 6.

 ISBN: 978-1-61208-388-9

We calculated the average percentage failure of nodes on every hour basis. Such a study can help in evaluating the behaviour of VM mechanisms. The failure percentage is calculated as:

$$failure\ (h) = \frac{numbr\ of\ failed\ nodes\ at\ \ h}{total\ number\ of\ nodes} * 100$$

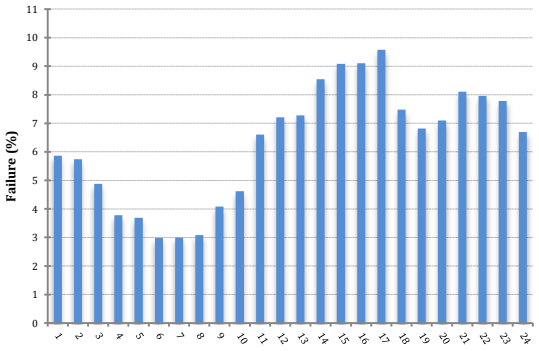


Figure 3. Average Hourly Failure

Figure 3 shows an average hourly failure percentage in 24 hour-period for analysis of 6 months run times of NotreDame nodes. The period is set to 24 hours because this is the running time set for our experiments. NotreDame failure analysis shows that failure percentage is about 3% as minimum in hour 6. Hour 17 recorded the highest failure percentages at about 10%. It is worth mentioning that on average about 6.3% of running nodes failed in an hour during the 6-month period. However, it was recorded that the percentage of node failures can reach up to 80% in some hours. This can demonstrate that failure events in Desktop Clouds are norms rather than exceptions.

B. Experiment Setting

The experiment is run for 180 times, each time representing a simulation of running NotreDame Desktop Cloud for one day. The run time was set to one day because the FTA provides a daily trace for NotreDame nodes as mentioned above. Each VM allocation mechanism is run for 180 times representing traces of 6 months from the FTA. This makes the total number of runs 540 (3 * 180). The workload was collected from the PlanetLab archive. The archive provides traces of real live applications submitted to the PlanetLab infrastructure [26]. One day workload was retrieved randomly as input data in this experiment. Each task in the workload is simulated as a Cloudlet in the simulation tool. The workload input remains the same during all the experiment runs because the aim of this experiment is to study the impact of node failures on throughput of Desktop Clouds.

The FTA files provide the list of nodes along with timestamps of failure/alive times. However, the specifications of nodes are missing. Therefore, we had set specification up randomly for physical machines. The missing specifications are technical specifications such as CPU power, RAM size and hard disk size.

Clients requested that 700 instances of VMs to run for 24 hours. There are four types of VM instances: *micro, small, medium and large*. They are similar to VM types that are offered by Amazon EC2. The type of each requested VM instance is randomly selected. The number of requested VMs and types remain the same for all run experiment sets. Each VM instance receives a series of tasks to process for a given workload. The workload is collected from PlanetLab archive which is an archive containing traces. PlanetLab is a research platform that allows academics to access a collection of machines distributed around the globe. A one day workload of tasks was collected using CoMon monitoring tool [27]. The same workload is submitted in every one day run.

In the experiment, if a node fails then all hosted VMs will be destroyed. The destruction of a VM causes all running tasks on the VM to be lost which consequently affect the throughput. The lost VM is started again on another PM and begins receiving new tasks. During running time, a node can become alive and rejoin the Cloud according to the used failure trace file. The simulation was run on a Mac i27 (CPU = 2.7 GHz Intel Core i5, 8 GB MHz DDR3) running OS X 10.9.4. The results were analysed using IBM SPSS Statistics v21 software.

V. RESULTS AND DISCUSSION

Table I shows a summary of descriptive results obtained when measuring the throughput output for each VM allocation mechanism implemented in NotreDame Cloud. N in the table means that the number of days is 180 days representing a six-month period. Kolmogorov-Smirnov (K-S) test of normality shows that the normality assumption was not satisfied because the FCFS and Greedy mechanisms are significantly non-normal, $P < .05$. Therefore, the non-parametric test Friedman's ANOVA was used to test which mechanism can yield better throughput. Friedman's ANOVA test confirms that throughput varies significantly from one mechanism to another, $X_F^2(3) = 397.14, P < .001$. Mean, median, variance and standard deviation are reported in Table I.

TABLE I. THROUGHPUT RESULTS

Mechanism	N	Mean	Median	Var.	St. Dev.	K-S Test
FCFS	180	79.21 %	78.77 %	37.03	6.09	$P < .05$
Greedy	180	88.61 %	89.48 %	16.85	4.1	$P < .05$
RoundRobin	180	85.47 %	85.29 %	15.13	3.89	$P = .2$

Three Wilcoxon pairwise comparison tests were conducted to find out which mechanism had the highest throughput. Note that three tests are required to compare three pairs of mechanisms which are FCFS Vs. Greedy, FCFS Vs. RoundRobin and Greedy Vs. RoundRobin mechanisms. The level of significance was altered to be 0.017 using Bonferroni correction [28] method because there

 ISBN: 978-1-61208-388-9

were 3 post-hoc tests required (.05/3 ≈ .017). The tests show that there is a significant difference between each mechanism with its counterpart. Therefore, it can be concluded that the Greedy mechanism yields the highest throughput since it has the median with highest value (median = 89.48%).

The median throughput of FCFS was about 79%, as being the worst mechanism among the tested mechanisms. Our findings confirm our expectation in section III.C. The RoundRobin came second in terms of throughput because the mechanism distributes load equally. So, node failures are ensured to affect the throughput. The median throughput was about 89% when Greedy VM mechanism was employed. The mechanism aims at maximising utilisation by packing as many VMs as possible to the same PM, thus reducing the number of running PMs. The average failure rate in submitted tasks is about 12%, given the average node failure percentage is about 6% as section IV.A shows. Such figures demonstrate the importance to develop fault tolerant VM mechanisms.

VI. RELATED WORK

Several simulators have been published to simulate Grid computing. SimGrid [16] is one of the early simulation tools to simulate Grid environment. GridSim [14] is another tool that fits within the same goal. CloudSim is built on top of GridSim. Donassole et al. [29] extended SimGrid to enable simulating Desktop Grids. Their work enables building a Grid on top of resources contributed by the public. The simulation tool is claimed to be of high flexibility and enable simulating highly heterogonous nodes. GroudSim [17] is a scalable simulation tool to simulate both Grid and Cloud platforms. The tool lets researchers to inject failures during running time. However, all of these tools fall short in providing virtualisation feature which is essential to evaluate VM allocation mechanisms.

WorkflowSim [19] is a new simulation extension that has been published recently as an extension for CloudSim tool. The tool was developed to overcome the shortage of CloudSim in simulating the scientific workflow. The authors add a new management layer to deal with overhead of complex scientific computational tasks. The authors argue that CloudSim fails in simulating the overhead of such tasks. The overhead may include queue delay, data transfer delay clustering delay and postscript. This issue may affect the credibility of findings and results. They, also, point out the importance of failure tolerant mechanisms in developing task scheduling techniques. WorkflowSim focuses on two types of failures: tasks failure and job failure. A Task contains a number of jobs, so a failure in a task causes a series of jobs to fail. However our work differs from WorkflowSim in the failure event and its impact. We focus on infrastructure level which contains nodes that host VMs whereas the authors are interested in the service level, i.e., the tasks and applications. We argue that service providers should consider developing failure tolerant mechanisms to overcome failures events in the infrastructure level.

DynamicCloudSim [18] is another extension for CloudSim tool. The authors are motivated by the fact that CloudSim lacks the ability to simulate instability and dynamic performance changes in VMs during runtime. This can have a negative impact on the outcome of computational intensive tasks which are quite sensitive to behaviour of VMs. The tool can be used to evaluate scientific workflow schedulers taking in consideration the variance of VM performance. In addition, execution time of a given task is influenced by I/O-bound such as reading or writing data. The authors extend instability to include tasks failure. Performance variation of running VMs is an open research challenge, but it is out of this scope.

VII. CONCLUSION AND FUTURE WORK

Desktop Cloud represents a new direction in Cloud computing. Desktop Cloud aims at exploiting idle computing resources to provide Cloud services mainly for research purposes. The success of Desktop Grids in providing Grid capabilities has stimulated the idea of applying the same concept within Cloud computing. However, Desktop Clouds use infrastructure that is very volatile since computing nodes have high probability to fail. Such failures can be problematic and cause a negative impact on the throughput of Desktop Clouds.

This paper presented a DesktopCloudSim as an extension tool CloudSim, a widely used Cloud simulation tool. DesktopCloudSim enables the simulation of node failures in the infrastructure of Cloud. We demonstrated that the tool can be used to study the throughput of a Desktop Cloud using NotreDame real traces. We showed that Greedy VM mechanism yielded better throughput in the presence of failures compared to the FCFS and RoundRobin mechanism.

The results of experiments demonstrate that node failures affect negatively the throughput outcome of Desktop Clouds. This opens a new direction to design a fault tolerant mechanism for Desktop Cloud. We intend to develop such a mechanism and evaluate it using the proposed tool. In addition, several metrics, such as power consumption and response time, should be used to evaluate VM mechanism.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Futur. Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, Jun. 2009.

[2] B. M. Segal, P. Buncic, D. G. Quintas, D. L. Gonzalez, A. Harutyunyan, J. Rantala, and D. Weir, "Building a volunteer cloud," *Memorias la ULA*, 2009.

[3] A. Alwabel, R. Walters, and G. Wills, "A view at desktop clouds," in *ESaaSA 2014*, 2014.

[4] G. Kirby, A. Dearle, A. Macdonald, and A. Fernandes, "An Approach to Ad hoc Cloud Computing," *Arxiv Prepr. arXiv1002.4738*, 2010.

[5] A. Marinos and G. Briscoe, "Community Cloud Computing," pp. 472–484, 2009.

[6] A. Andrzejak, D. Kondo, and D. P. Anderson, "Exploiting non-dedicated resources for cloud computing," *2010 IEEE Netw. Oper. Manag. Symp. - NOMS 2010*, pp. 341–348, 2010.

[7] A. Marosi, J. Kovács, and P. Kacsuk, "Towards a volunteer cloud system," *Futur. Gener. Comput. Syst.*, Mar. 2012.

[8] A. Gupta and L. K. L. Awasthi, "Peer enterprises: A viable alternative to Cloud computing?," in *Internet Multimedia Services Architecture and Applications (IMSAA), 2009 IEEE International Conference on*, 2009, vol. 2, pp. 1–6.

 ISBN: 978-1-61208-388-9

[9] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster Comput.*, vol. 12, no. 1, pp. 1–15, Oct. 2008.

[10] C. Gong, J. Liu, Q. Zhang, H. Chen, and Z. Gong, "The Characteristics of Cloud Computing," *2010 39th Int. Conf. Parallel Process. Work.*, pp. 275–279, Sep. 2010.

[11] I. Foster, Y. Zhao, I. Raicu, and S. Lu, "Cloud computing and grid computing 360-degree compared," in *Grid Computing Environments Workshop, 2008. GCE'08*, 2008, pp. 1–10.

[12] S. Choi, H. Kim, E. Byun, M. Baik, S. Kim, C. Park, and C. Hwang, "Characterizing and Classifying Desktop Grid," in *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*, 2007, pp. 743–748.

[13] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," *High Perform. Comput. Simulation, 2009. HPCS'09*, pp. 1–11, Jun. 2009.

[14] R. Buyya and M. Murshed, "Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing," *Concurr. Comput. Pract. …*, vol. 14, no. 13–15, pp. 1175–1220, Nov. 2003.

[15] F. Howell and R. McNab, "SimJava: A discrete event simulation library for java," *Simul. Ser.*, 1998.

[16] H. Casanova, "Simgrid: a toolkit for the simulation of application scheduling," *Proc. First IEEE/ACM Int. Symp. Clust. Comput. Grid*, pp. 430–437, 2001.

[17] S. Ostermann, K. Plankensteiner, R. Prodan, and T. Fahringer, "GroudSim: an event-based simulation framework for computational grids and clouds," *Euro-Par 2010 Parallel Processing Workshops*, no. 261585, pp. 305–313, 2011.

[18] M. Bux and U. Leser, "DynamicCloudSim: simulating heterogeneity in computational clouds," *SWEET '13 Proc. 2nd ACM SIGMOD Work. Scalable Work. Exec. Engines Technol.*, 2013.

[19] W. Chen and M. Rey, "WorkflowSim : A Toolkit for Simulating Scientific Workflows in Distributed Environments," 2012.

[20] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, D. Zagorodnov, and S. Barbara, "The Eucalyptus Open-Source Cloud-Computing System," *2009 9th IEEE/ACM Int. Symp. Clust. Comput. Grid*, pp. 124–131, 2009.

[21] J. Fontán, T. Vázquez, L. Gonzalez, R. S. Montero, and I. M. Llorente, "OpenNEbula: The open source virtual machine manager for cluster computing," in *Open Source Grid and Cluster Software Conference – Book of Abstracts*.

[22] B. Sotomayor, R. R. S. Montero, I. M. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," *IEEE Internet Comput.*, vol. 13, no. 5, pp. 14–22, Sep. 2009.

[23] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurr. Comput. Pract. Exp.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.

[24] B. Javadi, D. Kondo, A. Iosup, and D. Epema, "The Failure Trace Archive: Enabling the comparison of failure measurements and models of distributed systems," *J. Parallel Distrib. Comput.*, vol. 73, no. 8, pp. 1208–1223, Aug. 2013.

[25] B. Rood and M. J. Lewis, "Multi-state grid resource availability characterization," *2007 8th IEEE/ACM Int. Conf. Grid Comput.*, pp. 42–49, Sep. 2007.

[26] L. Peterson, S. Muir, T. Roscoe, and A. Klingaman, "PlanetLab Architecture : An Overview," no. May, 2006.

[27] K. Park and V. S. Pai, "CoMon," *ACM SIGOPS Oper. Syst. Rev.*, vol. 40, no. 1, p. 65, Jan. 2006.

[28] A. Field, *Discovering statistics using SPSS*, Third. SAGE Publications Ltd, 2009, p. 856.

[29] B. Donassolo, H. Casanova, A. Legrand, and P. Velho, "Fast and scalable simulation of volunteer computing systems using SimGrid," in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing - HPDC '10*, 2010, p. 605.

 ISBN: 978-1-61208-388-9

Optimized Cloud Resources Management Based on Dynamic Scheduling Policies and Elasticity Models

Nikolaos Chelmis, Dimosthenis Kyriazis, Marinos Themistocleous
Department of Digital Systems
University of Piraeus
Piraeus, Greece
e-mail: {helnik, dimos, mthemist}@unipi.gr

***Abstract*—Nowadays, the interest on Cloud Computing as a technical and business best practice has grown to great length. There is a huge pool of available cloud applications and services offered to end users. As application requirements, reflected to resources requirements (i.e., network, storage, computing capacity), are set by the application provider, a key issue relates to the resulting elasticity needs and their modeling. In addition to elasticity needs, application providers aim to maximize their customer base while considering the associated costs. To this end, business models are needed in order to attract customers while considering cost constraints. Their aim is to optimize the performance of the "investment" for resources compared to the expected number of customers. Nevertheless, the latter is directly linked to the provided quality of service and users' quality of experience. To this direction, in this paper, we present a mechanism that dynamically maps scheduling policies with the planned and estimated resources based on varying needs.**

Keywords-Cloud Computing; Infrastructure as a Service (IaaS); Elasticity; Monitoring; Scheduling Policy; Quality of Service.

I. INTRODUCTION

Cloud computing as a whole has rapidly evolved and became one of the most challenging paradigms of Information Technology. It has gained popularity for its ability to enable fast and effective access to large pools of virtualized resources and services that are dynamically provisioned to adjust to variable workloads and usage optimization [1]. This pool of resources is typically exploited by a pay-as-you-go [2] pricing model with the cost of using a cloud asset depending on the resources consumed. To this direction, cloud computing offers mechanisms to automatically scale applications in order to meet user's needs, thus making possible for them to rapidly adapt their resources to the workload minimizing the cost of overprovisioning.

There are three main classes in the cloud services stack which are generally agreed upon [1]: (i) Infrastructure as a Service (IaaS) where the provider sells access to computers upon which any software can run. The resources, which are in most cases virtual, are expressed in terms of processing power, memory, storage capacity, etc. (ii) Platform as a Service (PaaS), where an environment for application developers to deploy their code is offered. (iii) Software as a Service (SaaS) where customers pay to use an application that is hosted on a remote provider. The service provider manages the software and the underlying infrastructure. The main focus of this paper is the IaaS layer since it has great potential in further revolutionizing the way compute resources are provisioned and consumed.

While scalability enables smooth application execution even when number of users grows, two approaches are mainly used to make new resources available [2]: (i) Vertical scalability (scale up/down) increases or decreases the resources (commonly the CPU number, the memory or bandwidth) of an element in the system. (ii) Horizontal Scalability (scale in/out) replicates or removes instances of system elements (usually Virtual Machines - VMs) to balance the workload. Elasticity is the ability to scale an infrastructure on demand within minutes (seconds in an optimum case) to avoid under-utilization and over-utilization of resources. Scalability is a prerequisite for elasticity, but it does not take into consideration how fast or how often scaling actions can be performed thus it is not directly related to how well the actual resource demands are matched by the provisioned resources an any point of time [4].

What is more, modern business trends highlight opportunities for service provisioning via cloud infrastructure to the end users. Three main models have prevailed: (i) direct service provision to the end user (e.g., Dropbox), (ii) use of cloud infrastructure for an organization's internal purposes (e.g., internal network of a bank) and (iii) use of cloud infrastructure to provide a service to the end users.

The current paper focuses on the third model, which is being exploited by application providers / owners - referred as brokers. The service provided may be any software system, consisting of one or more components. Its architecture, in terms of components, interfaces and logic, is considered to be known and can be precisely described by the provider. As application requirements, regarding the resources, are formulated by the aforementioned brokers, a key issue relates to the resulting elasticity needs and their modeling. Elasticity and requirements are dependent on the following parameters: (i) application's nature (i.e., use of multiple processors), (ii) usage (i.e., variable exponential growth of end-users) and (iii) infrastructure vendors (i.e., resource availability). In order to analyze elasticity requirements, models that meet the above parameters and allow use of the analysis results to provide resources based on demand are required.

In addition to elasticity issues, application providers / owners aim to maximize their customer base while considering the cost. Towards this direction, dynamic scheduling policies that suggest ways to attract customers are required, with an ultimate goal to optimize the "investment" made for resources compared to the foreseen number of customers. Nevertheless, the expected number is directly linked to the Quality of Service (QoS) provided and users' Quality of Experience (QoE). For example, if provisioned

 ISBN: 978-1-61208-388-9

resources follow demand, a number of users will have to wait for resource's availability and hence the use of service.

In economic terms, the overprovisioned resources are easily measured but underprovisioned is way harder. End users who were unable to use the application or experienced performance degradation (lower levels of QoS) may never become returning customers and discredit the service. Based on the above, a dynamic model, efficient for business, regarding both current users and expected ones compared to elasticity needs, is required. Nonetheless, the efficiency of scheduling policy needs to be linked to the required resources that accommodate the users' expectation regarding QoS. The linkage / mapping should allow elasticity models to be followed compared to different scheduling policies (aggressive, passive, neutral) resulting in the optimal resource management. Furthermore, scheduling policies should be able to be switched a dynamic way during runtime. In this paper, we present a mechanism (overview depicted in Figure 1) enabling the latter, which has been developed and validated in the framework of the PinCloud project [5] with different stakeholders in the eHealth domain.

A mechanism, the overview of which is presented in Figure 1, enabling the latter is presented in this paper.

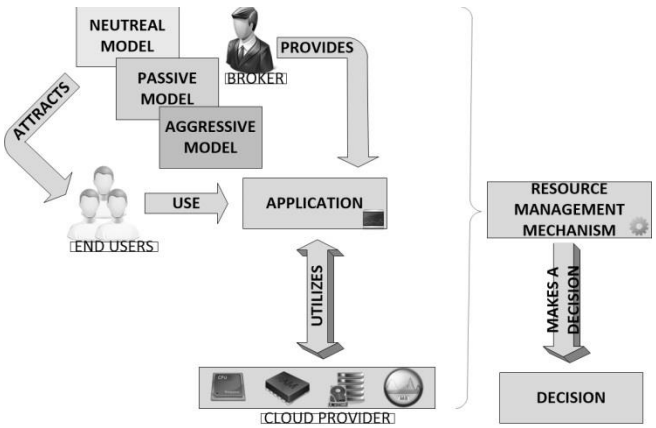


Figure 1. Use of Mechanism

The added value of the proposed mechanism lies on the incorporation of techno-economic factors for the management of resources in cloud environments. The mechanism enables the outcomes of a business - expressed through the corresponding models - simulation process to be considered during runtime in order to trigger resource provisioning decisions. Runtime adaptation takes place based on both the business goals of the application provider and the emerging requirements from the end-users.

The remainder of this paper is structured as follows: Section 2 presents the related work that solves the problem of elasticity in cloud platforms. Section 3 proposes three dynamic scheduling policies, while in Section 4 the architecture of the proposed mechanism's architecture is being analyzed. In Section 5, mechanism is evaluated and the results are discussed. Finally, Section 6 concludes our work and discusses open areas.

II. RELATED WORK

Elasticity, a term originally defined in physics, is considered one of the central attributes of the cloud. Cloud providers use the term in advertisements and even in the naming of products or services and implement it in different degrees.

Amazon EC2 [6] allows VMs to scale vertically in order to reciprocate with resource requirements. Customers can change resource requirements; but, in order to achieve horizontal scalability a cluster of VMs must be created and configured according to needs. This is a manual process, which does not include application configuration of a VM. Amazon EC2 enables the preparation of the VM but not the automatic configuration, which is a major requirement for an elastic platform.

Microsoft's Windows Azure [7] consists of three main components providing a set of services to cloud users for running applications and storing data. Azure offers specific VM instances with predefined sizes (CPU, Memory). Automatic scaling is offered through application rules via a configuration file specified by users.

Google App Engine [8] is optimized for web applications. It handles the deployment, monitoring and launching of service instances making use of Google's core engine. Automatic scalability is transparent to the application providers / owners with no option for the developer to write his own scaling rules based on application's specific needs.

Amazon offers a service called Spot Instances [9], an elasticity solution based on cost. Spot Instances are virtual servers sold per hour via an action. Based on bids and available capacity Amazon determines a price (Spot Price) and if the maximum bid price exceeds the current Spot Price the request is fulfilled.

RightScale [10] is an application management platform for clouds addressing the Infrastructure-as-a-Service (IaaS) service model. RightScale provides control and elasticity capabilities, assisting the user to design, deploy and manage applications on a number of underlying clouds (i.e., Amazon, Rackspace, private solutions like CloudStack, OpenStack and others). Various monitoring metrics are used and users can define alerts based on these metrics.

OnApp [11] is a software package for IaaS cloud providers. It states that it enables replication and redimensioning on VMs allowing changes manually or automatically, based on rules defined by user and metrics obtained by the monitoring mechanism. Lim et al. [12] proposed an automatic mechanism based on a target range for a specific system metric, rather than a threshold to trigger actions. The key point is that the system reacts when the defined metric is outside the range, reducing resources allocations.

Internal provider resources management and use of elasticity is addressed by Meng et al. [13]. According to the authors, management tasks (like VM creation, migration, etc.) are expensive in terms of computation and more likely to occur in bursts. Lack of resources to handle this workload will affect users' applications performance. Based on this observation, TIDE: a self-scaling framework for virtualized data center management, was proposed. The main idea is to treat management workload the way application workload would be treated. When bursts in management workload are encountered by TIDE, it powers up dynamically additional server management instances. When burst subsides, management instances' physical resources can be used for user application workloads.

Comparing to the approaches discussed above, the proposed resource management mechanism addresses the

 ISBN: 978-1-61208-388-9

issues of resource management from a broker's perspective. To take full advantage of elasticity, while an elastic infrastructure provides the required functionality, business adoption is constrained by the associated costs compared to the actual and foreseen service usage. Applications should have the ability to dynamically exploit the infrastructure according to workload changes in a dynamic and cost-efficient way. The presented mechanism proposes resource management (in terms of resource requirements specification) taking into consideration both the forecasted elasticity needs of the service in relation with the scheduling policy being followed by the service owner.

III. SCHEDULING POLICIES

Application providers / owners use different business models in order to maximize their profit. Among other parameters (like cost for supplies, man hours etc.) these models also include scheduling policies which take into consideration the resources that application needs to operate inside the predefined QoS. Dynamic scheduling policies reflect how the current and the forecasted number of users relate to the application needs (and thus resource requirements) for specific predefined QoS levels. Given that the provided service can be charged based on different models, three main policies are proposed (i) Aggressive, (ii) Passive, and (iii) Neutral. Each model guarantees different response time, therefore aims to different customer base size. Concept of man hours, cost for supplies, and others are out of the scope of this work; therefore, they are not taken into consideration. The core set of parameters incorporated in the considered scheduling policies follow:

1. Cloud Resources
   (a) Type (CPU, Memory)
   (b) Availability
2. Usage
   (a) Number of users currently using application
   (b) Target users
   (c) Number of acquired users
3. Cost
   (a) Cost of acquiring resources
   (b) Gain from users acquired

The above parameters are directly and dynamically linked / related to each other. For instance, increase of users means more revenue but it also means increase in response time. In order for the response time to be maintained within specific limits, additional resources may need to be acquired. Accordingly, decrease in the number of users means lower response time, reducing the need for resources.

In order to create the dynamic scheduling policies, the concepts of minimum (t^L) and maximum (t^U) response time are introduced. Response time should never exceed any of these two limits. Another key element taken into consideration, which can alter application's response time, is spin-up time [14]. The amount of time needed, since initial acquisition request, for required resources to be ready for use is called spin-up time. Weighting factors are included in the scheduling policies in order to define the requested amount or resources. All the attributes used for the creation of the scheduling policies are summarized in Table 1.

TABLE 1. ATTRIBUTES USED FOR SCHEDULING POLICIES

Component	Description
Goal Users	Number of target Users
RT	Response Time
t^L	Best – Minimum Response Time
t^U	Worst – Minimum Response Time
t^{SU}	Spin Up Time
initDep	Initial Deployment
res	Resources
a,b,c	Weighting Factors
n	Predifined amount of time

Based on the above, the following paragraphs present the scheduling policies taken into account in the current work: (i) Passive, (ii) Neutral, and (iii) Aggressive. For each one, a mathematical equation describes how the new value for the resources (*f(x)*) is calculated.

A. Passive Scheduling Policy

This model ensures that users do not experience violation of the maximum response time but allows response times to be close to the maximum ones. To achieve that the difference between current response time and minimum response time plus a predefined amount of time, referred as n, is examined and if current response time is higher more resources are requested. Bursting (i.e., extreme growth in user numbers) is also taken into consideration as current response time is contradicted to maximum response time. Spin-up time is not taken into consideration at this model. Finally, to avoid overprovisioning current response time is compared to minimum response time and if it is less then all acquired resources are released. This model is described in (1) and its flowchart is illustrated in Figure 2.

$$f(x) = \begin{cases} a * res, & RT > t^L + n \\ b * res, & RT \geq t^U \\ initDep, & RT < t^L \end{cases} \quad (1)$$

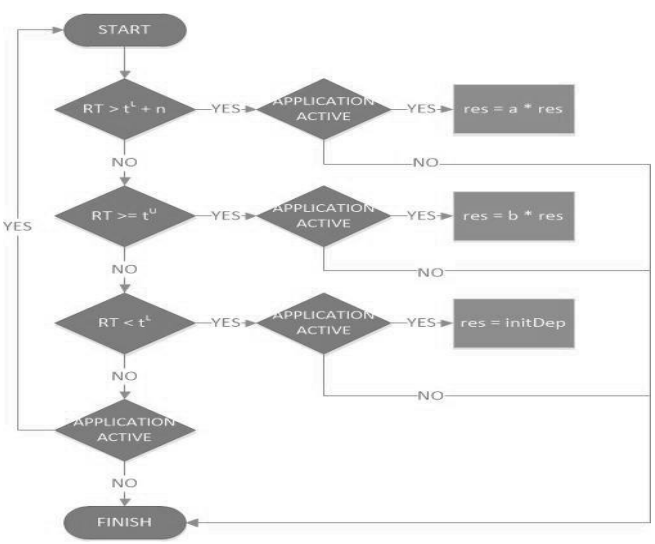


Figure 2. Passive Scheduling Policy

B. Neutral Scheduling Policy

This model's target is to prevent users, even in bursting, to come too close to maximum response time. It follows same logic as the above model comparing current response time with minimum and maximum. In contrast with passive model, this one takes into consideration spin-up time in cases of bursting. Spin-up time is subtracted from maximum response time, ensuring that the requested resources will be available before maximum response time is reached. Furthermore, if the number of target users is reached more resources are acquired as a bonus to users. Overprovisioning is avoided the same way as in passive model. This model is described in (2) and its flowchart is illustrated in Figure 3.

$$f(x) = \begin{cases} a * res, & RT > t^L + n \\ a * res, & Users = GoalUsers \\ b * res, & RT \geq t^U - t^{SU} \\ initDep, & RT < t^L \end{cases} \quad (2)$$

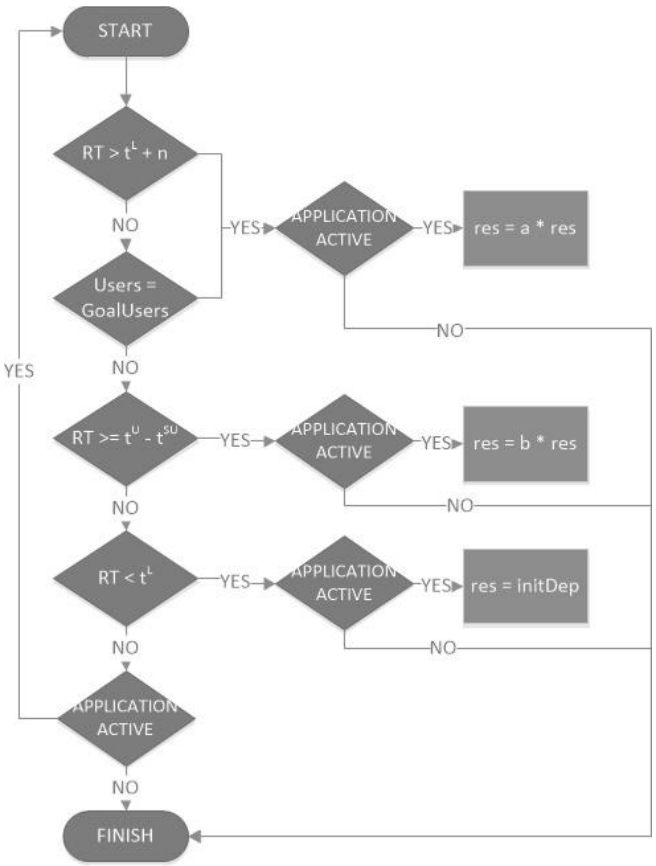


Figure 3. Neutral Scheduling Policy

C. Aggressive Scheduling Policy

This model makes sure users are as close as possible to minimum response time. It is similar to Neutral model although it's weighting factors are bigger. Furthermore, a predefined amount of time (n) is added to spin-up time, ensuring that even in heavy bursts users will not reach close to the maximum response time. Overprovisioning is again avoided by comparing application's response time with minimum response time. This model is described in (3) and its flowchart is illustrated in Figure 4.

$$f(x) = \begin{cases} a * res, & Users = GoalUsers \\ b * res, & RT \geq t^L + n \\ c * res, & RT \geq t^U - (t^{SU} + n) \\ initDep, & RT < t^L \end{cases} \quad (3)$$

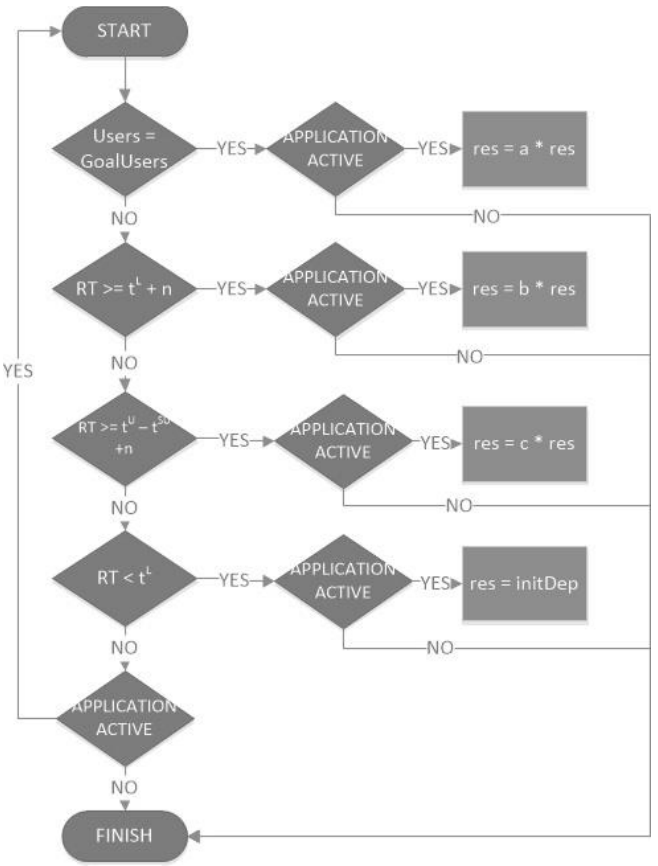


Figure 4. Aggressive Scheduling Policy

The aforementioned policies are supposed to be part of an application's provider / owner business model. According to what QoS application provider / owner promises on his business model the corresponding policy should be followed. Furthermore, weighting factors and the predefined amount of time (n) are different in each policy. That means that weighting factors and predefined amount of time of passive scheduling policy are the lowest while aggressive one's are the bigger.

IV. RESOURCE MANAGEMENT MECHANISM

The goal of the mechanism is to make it completely independent to the process of starting the application, thus making it possible to start an application and join the mechanism later if needed. Secondly, the mechanism should allow application providers /owners to change / switch between scheduling policies in a dynamic way during runtime.

 ISBN: 978-1-61208-388-9

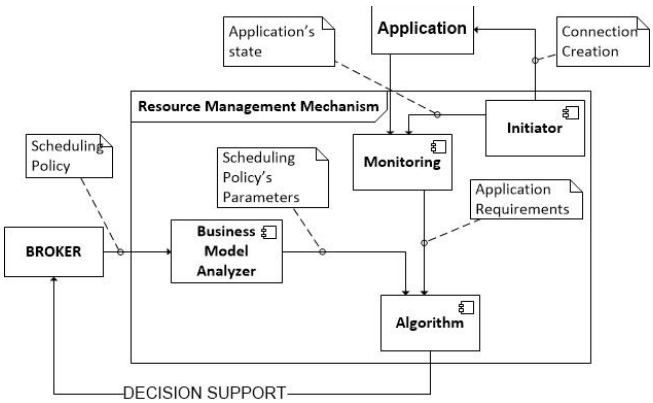


Figure 5. Mechanism's Components

As depicted in Figure 5, the proposed modular architecture consists of four main building blocks / services:

1. **Initiator**: This service creates a connection with the application and obtains its current requirements and current usage (e.g., number of users, number of requests, etc.). After relaying this information to the monitoring component it pauses.
2. **Monitoring**: The goal of this service is to monitor the application's state in terms of both application-level (e.g., number of current users and current response time) and resource-level (e.g., CPU usage) metrics. All the information is passed to the algorithm.
3. **Business Model Analyzer**: The analyzer obtains the current scheduling policy, contained in the business model used by the provider and relays the information to the algorithm. Furthermore, number of target users (referred as goal users in the above section) is also relied to the algorithm.
4. **Algorithm**: Mechanism's logic which takes into consideration both the forecasted elasticity needs of the service and the scheduling policy which the service owner follows. Based on the collected information from the aforementioned components it estimates response time for the acquired users. Since business models, thus scheduling policies, can be changed by application provider / owner during runtime number of target users can also be changed. Each time number of target users is reached application provider / owner can set a new goal number and the mechanism will estimate the response time.

V. EVALUATION

The aim of experimentation is to evaluate the proposed mechanism in a real environment. To this end, the experiment was distributed across three different locations in Europe: EPCC (Edinburgh), HLRS (Stuttgart) and PSNC (Poznan) provided by the BonFire cloud infrastructure [15]. The connection between individual sites was over best effort Internet (Cloud over Internet) besides the connection between UK-EPCC and PL-PSNC sites. The latter was established with GÉANT Bandwidth on Demand (BoD) system (AutoBAHN BoD version 2.1.1[16]), which is a service for dynamic bandwidth provisioning across multiple networks (guaranteed bandwidth).

The experimentation infrastructure that had been used consisted of in total of 30 VMs acting as servers (10 VMs have been deployed in each of the following sites: Edinburg, Stuttgart and Poznan). Given that the goal of the epxerimenation was to obtain information with respect to response times, the clients have been deployed in 60 VMs in different sites so as to obtain information for cross-site response times. The response time was measured through Apache JMeter.

For the purposes of the experimentation, a simple Java servlet-based Service Oriented Architecture (SOA) service was developed. Upon receiving an HTTP GET request, the service calculates a million random integers ranging from zero to one thousand. Application does not store user state, thus making easy the service requests to be spread among servers running the same code. The only purpose of the service is to represent highly parallelizable computation task. Requests arrive at a load balancer node, which is included to the deployment, to allocate them to servers. Application was deployed in each client (side).

A. Evaluation Results

The aim of our evaluation was to validate the operation and efficiency of the algorithm in different cases and for different metrics. In the experiment presented below, a total of 200 users were used, while policies of neutral scheduling policy where applied with minimum response time set to 200 and maximum set to 15000 milliseconds.

The experimentation data that have been collected are depicted in Figure 6. The information includes active users during the experiment period, response time, CPU and memory utilization. It can be observed that number of users was increased linearly. Response time though was not increasing with the same pattern since it is also dependent on CPU and memory usage. The increase on number of users entailed increase on resources; thus, the response time was maintained steady. As number of users kept on increasing response time started to increase leading to more CPU usage. One should take into consideration application's nature: It requires computation power but not memory usage since it requires no cashing. As observed in Figure 6 memory has an increase at the beginning of the experiment, but then remains steady, while CPU resources are increased, proving that resource management was correct. Response time contradicted, only with the number of users is illustrated in Figure 7.

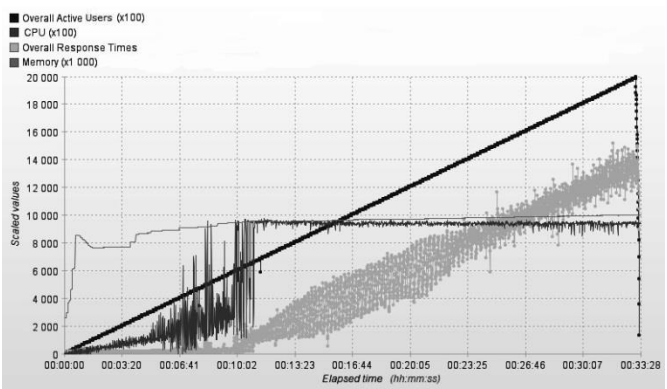


Figure 6. Collected Experimentation Data

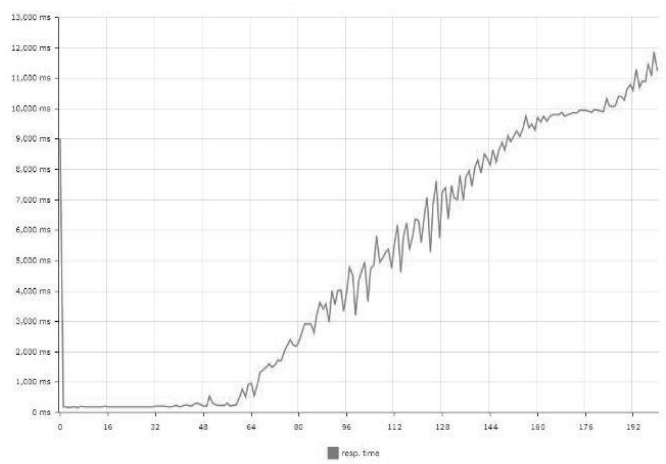


Figure 7. Response Time VS users

The distribution of response times during the evaluation experiment is illustrated in Figure 8. With a closer observation, one can notice that, regardless the linear growth of users, the response time was not also growing linearly but was maintained steady for an amount of time. Another important observation is that the distribution of lower response times (i.e., 300-900ms) is enough smaller. That is easily explained: Since neutral's model policies were applied as long as users were closer to minimum response time no resources were acquired so with number of users growing response time also grew.

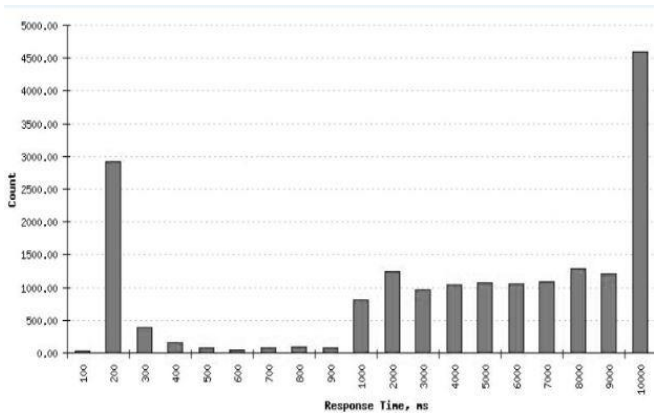


Figure 8. Response Time Distribution

Figure 9 presents a comparison between estimated and actual response time. As illustrated in the figure, the proposed mechanism delivers results very close to reality. As a result, the mechanism identifies the forecasted number of users in an accurate way as required to state the resources required for the number of users.

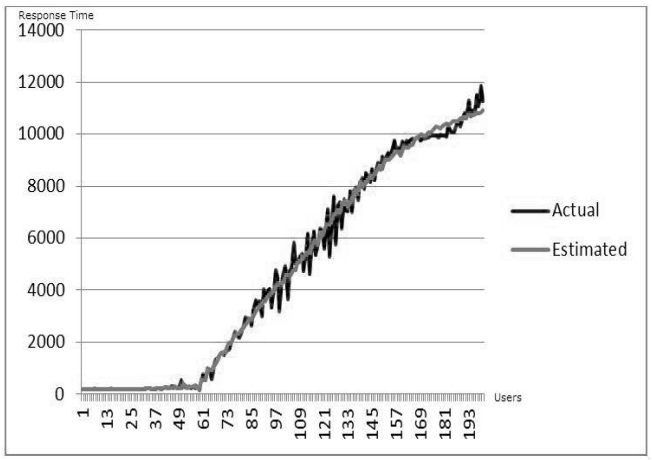


Figure 9. Actual VS Estimated Repsonse Time

In order to illustrate more clearly whether the actual coincides with the estimated response time in Figure 10, a histogram of residuals resulting from the comparison between them is illustrated.

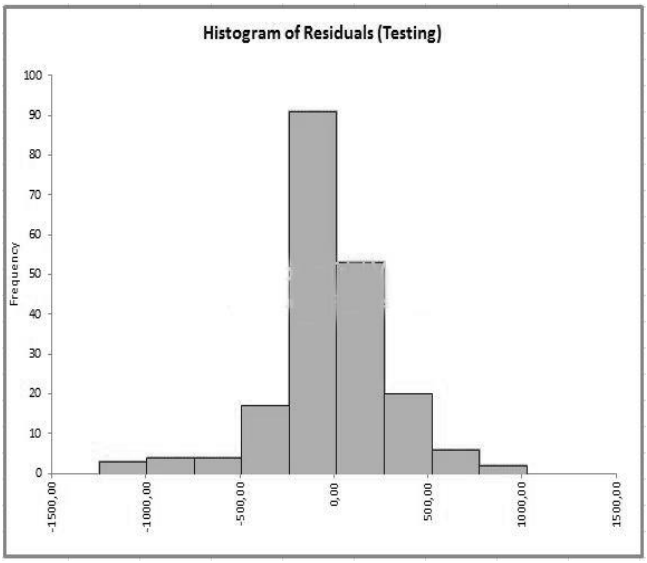


Figure 10. Histogram of Residuals

As shown, the highest residues occur around zero, thus proving the correctness of the prediction results. It is, therefore evident that the mechanism determines the number of users compared to the required resources modeled with precision.

VI. CONCLUSION AND FUTURE WORK

The objective of this paper was to present a new mechanism for the optimum cloud resources management based on dynamic scheduling policies and elasticity needs. Initially, dynamic scheduling policies regarding both current users and expected ones compared to elasticity needs were proposed. Application's response time was compared to the maximum and minimum accepted response time defined in the QoS. Furthermore, a mechanism that maps current and expected users with resource needs while taking into consideration the scheduling policies was introduced. The proposed mechanism is completely independent from the application and can be deployed while an application is

 ISBN: 978-1-61208-388-9

active. Service provider can change his scheduling policy during runtime without affecting neither application's nor mechanism's performance.

ACKNOWLEDGMENT

The research leading to the results presented in this paper has received funding from the European Union and the Greek National Strategic Reference Framework Programme (NSRF 2007-2013), Project PinCloud under grant agreement number "11SYN_6_1013_TPE".

REFERENCES

[1] Grance, P.M.a.T., NIST Definition of Cloud Computing, Version 15. 2011, NIST: http://csrc.nist.gov/groups/SNS/cloud-computing. [retrieved: January 2015]

[2] Suleiman, B., Sakr, S., Venugopal, S., and Sadiq, W. (2012), "Trade-off analysis of elasticity approaches for cloud-based business applications", In Web Information Systems Engineering-WISE 2012 (pp. 468-482). Springer Berlin Heidelberg.

[3] Rimal, B. P., Choi, E., and Lumb, I. (2009, August), "A taxonomy and survey of cloud computing systems", In INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on (pp. 44-51). Ieee.

[4] Herbst, N. R., Kounev, S., and Reussner, R. (2013, June), "Elasticity in Cloud Computing: What It Is, and What It Is Not", In ICAC (pp. 23-27).

[5] PinCloud Project. Available from: http://pincloud.med.auth.gr/en. [retrieved: February 2015]

[6] Amazon. Amazon EC2. Available from: http://aws.amazon.com/ec2/. [retrieved: January 2015]

[7] Microsoft Windows Azure. Available from: http://azure.microsoft.com/enus/documentation/. [retrieved: January 2015]

[8] Google App Engine. Available from: https://developers.google.com/appengine/?csw=1. [retrieved: January 2015]

[9] Amazon EC2 Spot Instances. Available from: http://aws.amazon.com/ec2/purchasing-options/spot-instances/. [retrieved: January 2015]

[10] RightScale. Available from: http://www.rightscale.com/. [retrieved: January 2015]

[11] onApp. Available from: http://onapp.com/. [retrieved: January 2015]

[12] Lim, H. C., Babu, S., Chase, J. S., and Parekh, S. S. (2009, June), "Automated control in cloud computing: challenges and opportunities", In Proceedings of the 1st workshop on Automated control for datacenters and clouds (pp. 13-18). ACM.

[13] Meng, S., Liu, L., and Soundararajan, V. (2010, November), "Tide: achieving self-scaling in virtualized datacenter management middleware", In Proceedings of the 11th International Middleware Conference Industrial track (pp. 17-22). ACM.

[14] Brebner, P. C. (2012, April), "Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (iaas) cloud applications", In Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering (pp. 263-266). ACM.

[15] Kavoussanakis, K., et al., "Bonfire: The clouds and services testbed", In Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on (Vol. 2, pp. 321-326).

[16] Bonfire Documentation – Controlled Bandwidth with AutoBAHN. Available from: http://doc.bonfire-project.eu/R3.1/networking/autobahn.html. [retrieved: January 2015]

 ISBN: 978-1-61208-388-9

Fragmentation-Aware Load-Balancing Virtual Optical Network Embedding (VONE) Over Elastic Optical Networks

Fariborz Mousavi Madani, Sheida Mokhtari
Faculty of Engineering
Alzahra University
Tehran, Iran
email: mosavif@alzahra.ac.ir, sh.mokhtari@student.alzahra.ac.ir

Abstract— **Optical network virtualization has recently been studied extensively as a promising solution to share physical infrastructure resources among different users and applications. Virtual network embedding over elastic optical networks has recently attracted massive attentions as a promising solution to realize fine-grained flexibility in resource provisioning. Variation of bandwidth requirements associated with virtual network requests accompanied by nondeterministic nature in arrival and holding times of them result in fragmentation of spectral blocks along both frequency and spatial directions. Spectral fragmentation refrains commitment to spectrum contiguity and spectrum continuity constraints in lightpath establishment thereby severely exacerbate blocking probability, as well as spectrum utilization. Fragmentation-aware routing and spectrum assignment problem has been extensively studied in the context of elastic optical network provisioning. These schemes, however, cannot be applied to virtual network embedding since we cannot refer to routing and spectrum assignment problem to implement virtual link mapping without regard to virtual node mapping. Therefore, in this paper, an integer linear programming formulation for fragmentation-aware virtual network embedding was developed for the first time to our knowledge. Numerical simulations for three different scenarios were carried on and demonstrated that our proposed model outperforms the previous work over a wide range of offered traffic loads under the same conditions.**

Keywords— elastic optical network; fragmentation-aware; load-balancing; VONE

I. INTRODUCTION

Optical networks are evolving from a fixed International Telegraph Union-Telecommunication (ITU-T) Dense Wavelength Division Multiplexing (DWDM) wavelength grid to a flexible grid in which the optical spectrum is divided into smaller Frequency Slots (FSs) with 6.25 or 12.5 GHz width each. In such a flexible grid or Elastic Optical Network (EON), resources are assigned by matching resource requirements of connection requests and the necessary number of FSs. The flexible use of the spectrum grid enabled by EONs promotes a more efficient and adaptable use of networks resources. However, the setup and release of connections in a dynamic network scenario can create gaps in the optical spectrum, which sizes in terms of the number of FSs may not be sufficient to accommodate an incoming connection request, causing what is known as the fragmentation problem. The spectrum fragmentation problem may lead to inefficient resource utilization and a high blocking probability [1]. A Virtual Optical Network Embedding (VONE), generally, consists of two stages, node mapping and link mapping, because of the complexity originated from correlation between these two stages, we brought it up in Integer Linear Programming (ILP) formulation. A Virtual Optical Network (VON) is composed of several Virtual Nodes (VNs) interconnected by Virtual Optical Links (VOLs). Typically, a network operator constructs VONs using optical network embedding, which allocates necessary resources in the physical infrastructure to each VON through node and link mapping stages. Due to the complex inter-dependence between these stages, VONE has become a major challenge for optical network virtualization [2]. Even though VONE over fixed-grid Wavelength-Division Multiplexing (WDM) networks, assuming that all the Substrate Nodes (SNs) were equipped with sufficient wavelength converters has been studied extensively [3]-[5], VONE over flexible-grid EONs has just started to attract research interests [2][6]-[8], which can potentially provide efficient support to emerging cloud services, especially for the highly distributed and data-intensive applications such as petabits-scale grid computing. In this paper, we develop and investigate an ILP model for Fragmentation-Aware Load-Balancing-VONE (FALB-VONE) over flexible-grid EONs for the first time to the best of our knowledge. The simulation results from three different scenarios demonstrate that our proposed ILP model outperforms previous work on minimization of resource utilization.

The rest of the paper is organized as follows. Section II surveys related work on VONE. The network models and problem descriptions of FALB-VONE over EONs are presented in Section III. Section IV discusses the ILP model and the performance evaluations of FALB-VONE. Finally, Section V concludes the paper.

II. RELATED WORKS

Most of the previous studies on VONE were targeted for fixed-grid WDM networks, but recent studies have suggested that optical network virtualization over flexible-grid EONs can potentially provide efficient support to emerging cloud services, especially for the highly distributed and data-intensive applications such as petabits-scale grid computing [2]. Chen et al. [9] investigated a spectrum allocation

 ISBN: 978-1-61208-388-9

approach through utilizing the relationship between accommodation capability of spectrum blocks and traffic bandwidth distribution. Based on this, a Fragmentation-Aware Spectrum Allocation (FSA) algorithm was presented. Zhang, Lu, Zhu, Yin, and Yoo [10] discussed two Routing and Spectrum Assignment (RSA) algorithms that can control the increase of fragmentation ratio and reduce necessary defragmentation operations in networks, the first one is the Minimum Network Fragmentation Ratio RSA (MNFR-RSA) algorithm, which makes resource allocation based on the Network Fragmentation Ratio (NFR) and the second one is the Maximum Local Utilization RSA (MLU-RSA) algorithm, which alleviated bandwidth fragmentation by utilizing the slots that have already been used the most in the network. In [11] Talebi, et al. examined and categorized solution approaches to Fragmentation-Aware RSA (FA-RSA) including Proactive FA-RSA, which attempts to prevent or minimize spectrum fragmentation at the time a new request is admitted and Reactive FA-RSA that employs defragmentation techniques to accommodate high-rate and long- path connections. The objective of defragmentation is to rearrange the spectrum allocation of existing traffic demands so as to consolidate available slots into large contiguous and continuous blocks. Yin, Zhu, and Yoo in [12] analyzed the fragmentation problem in detail for service provisioning in dynamic EONs, and proposed three fragmentation-aware algorithms for Routing, Modulation and Spectrum Assignment (RMSA) in EONs. Moura, Fonseca, and Scaraficci [13] introduced a multi-graph Shortest Path algorithm, which represented the spectrum occupancy by a multi-graph, where allocation decisions were based on cost functions, which tried to capture the potentiality of spectrum fragments to allocate incoming requests. In [14], Yin et al. have investigated the spectrum fragmentation problem in EON in both the spectral and the spatial dimensions and have proposed two fragmentation aware RSA algorithms to proactively prevent fragmentation during RSA to the incoming lightpath request.

Among the published papers on VONE, [2][8][15] have taken care of minimizing spectral utilization of substrate fiber links for mapping the requested virtual link bandwidth, possibly with the cost of imposing spectral fragmentation and intensifying load unbalancing. In the following section, we will develop an ILP formulation that encompass appropriate fragmentation-awareness and load-balancing expressions within the objective function.

III. MATHEMATICAL MODEL

We model the substrate EON as an undirected graph, denoted as $G^s(V^s, E^s)$, where V^s is the set of SNs, and E^s represents the set of Substrate Fiber Links (SFLs). We assume that each SN $v^s \in V^s$ has a computing capacity of $c^s_{v^s}$ and each SFL $e^s \in E^s$ can accommodate B^s Frequency Slots (FS').

Similar to the substrate topology, a VON request can also modeled as an undirected graph $G^r(V^r, E^r)$, where V^r is the set of virtual nodes (VNs), and E^r represents the set of VOLs. Each VN $v^r \in V^r$ is associated with a computing resource requirement $c^r_{v^r}$, while the Band Width (BW) requirement of each VOL $e^r \in E^r$ is $bw^r_{e^r}$ (in terms of Gb/s). This work considers both node mapping and link mapping in the VONE process. In the course of node mapping, each VN from the VON request is mapped onto a unique SN that has sufficient computing capacity. The link mapping course is essentially a special RSA operation. Specifically, the RSA sets up a lightpath in the physical infrastructure for each VOL to satisfy its BW requirement, under the spectrum non-overlapping, continuity and contiguity constraints [2].

The next subsection introduces a path-based ILP formulation for the FALB-VONE over EONs following the recent work presented in [2]. In-line with modulation-level adaptability provided by Optical Orthogonal Frequency Division Multiplexing (OOFDM) scheme, we assume that substrate lightpath can select one of Binary Phase-Shift Keying (BPSK), Quadrature Phase-Shift Keying (QPSK), 8 Quadrature Amplitude Modulation (8QAM), and 16QAM modulation-levels adaptively according to the transmission distance. If we take the BW of one FS to be 12.5 GHz, it can carry 12.5 Gb/s signal when the modulation-level is BPSK. Based on the experimental results in [16], we set the transmission reach of BPSK, QPSK, 8QAM, and 16QAM signals as 3000, 1500, 750, and 375 km, respectively. For all pre-calculated substrate shortest paths, the highest possible modulation-levels are predetermined and stored. BW requirement of VOL is normalized as multiples of one subcarrier BPSK signal operating at 12.5 Gbs.

A. Definitions

1) $Y \triangleq |E^r|$ Total number of VOLs in a VON request.
2) $M \triangleq |E^s|$ Total number of links in substrate EON.

B. Notations

1) $G^s(V^s, E^s)/G^r(V^r, E^r)$ Graph of physical infrastructure/ VON request.
2) $c^s_{v^s}/c^r_{v^r}$ Computing capacity/resource requirement of each SN $v^s \in V^s$/VN $v^r \in V^r$
3) B^s Total number of FSs on each SFL $e^s \in E^s$.
4) $w^s_{e^s,k}/z^s_{e^s,k}$ Starting/Ending FS index of kth Maximal Contiguous Slot-Block (MCSB), which is a Contiguous Slot-Block (CSB) that includes all the available and contiguous FS(s) at a spectral location on SFL e^s.
5) $s^r_{e^r}/d^r_{e^r}$ End-nodes of the VOL $e^r \in E^r$.
6) P^s Set of all pre-calculated routing paths in the substrate network $G^s(V^s, E^s)$.
7) $s^s_{p^s}/d^s_{p^s}$ Source/ Destination node of the path $p^s \in P^s$.
8) $P^s_{e^s}$ Set of routing paths that use the SFL e^s, obviously, $P^s_{e^s} \subset P^s$.
9) $m^s_{p^s}$ Highest modulation-level for path p^s.
10) $bw^r_{e^r}$ Normalized BW requirement of the VOL e^r in the VON request $G^r(V^r, E^r)$.

11) $n_{e^s}^s$ Number of MCSBs on SFL e^s before FS assignment.
12) $\eta_L^{min}/\eta_L^{max}$ Minimum/Maximum value of η_L ; weighting coefficient for link load balancing expression in the objective function.
13) $\eta_N^{min}/\eta_N^{max}$ Minimum/Maximum value of η_N ; weighting coefficient for node load balancing expression in the objective function.
14) $\Delta C_L^s/\Delta C_N^s$ Difference between maximum and minimum available Link/Node capacities before FS assignment.

C. Objective Function

Minimize

$$\frac{\alpha}{M}\cdot\sum_{e^s\in E^s}\left(\frac{\Psi_{e^s}}{w_{e^s,1}^s}\right)+\frac{\beta}{M}\cdot\sum_{e^s\in E^s}\left(\sum_{m=1}^{n_{e^s}^s+Y} f_{e^s,m}\Big/ n_{e^s}^s\right)$$
$$+\frac{\gamma}{\sum_{e^r\in E^r}\left\lceil bw_{e^r}^r\ /\ \max_{p^s\in P^s} m_{p^s}^s\right\rceil}\cdot\sum_{e^r\in E^r}\sum_{p^s\in P^s}\zeta_{e^r,p^s}\cdot|p^s|\cdot\left\lceil\frac{bw_{e^r}^r}{m_{p^s}^s}\right\rceil \quad (1)$$
$$+\eta_L\cdot\frac{C_L^{max}-C_L^{min}}{\Delta C_L^s}+\eta_N\cdot\frac{C_N^{max}-C_N^{min}}{\Delta C_N^s}$$

where, the first term is introduced to keep the assigned CSBs as close as possible to the starting index of first MCSB. It comprises sum of ratios of highest starting slot index among CSBs allocated to $e^r s$ that uses $e^s$to the starting FS index of the first MCSB of that e^s divided by M to get the normalized value. α is the weighting coefficient to adjust the relative contribution of this term. The second term is included to minimize extra spectral fragmentation induced by VOL mapping and consists of normalized sum of total number of free slot blocks in e^s after allocating CSBs to $e^r s$ divided by number of MCSBs of that e^s. β is the weighting coefficient to tune its impact. The third term accounts for minimizing BW utilization, likewise presented in [2], but here, it is divided by a lower band of BW utilization for normalization purpose and γ is the corresponding weight coefficient. Finally, the last two terms are responsible to maintain link and node load balancing in substrate EON. η_L and η_N are their weighting coefficients, respectively, which are computed from the given parameters $\eta_L^{min}/\eta_L^{max}$ and $\eta_N^{min}/\eta_N^{max}$ by the following equation:

$$\eta_L=\left(\eta_L^{max}-\eta_L^{min}\right)\cdot\exp\left(-minC_L^s/\Delta C_L^s\right)+\eta_L^{min}$$
$$\eta_N=\left(\eta_N^{max}-\eta_N^{min}\right)\cdot\exp\left(-minC_N^s/\Delta C_N^s\right)+\eta_N^{min} \quad (2)$$

Under excessive load unbalancing conditions, some links are highly loaded ($minC_L^s\approx 0$) while some others are lightly loaded ($\Delta C_L^s\gg 0$) so the ratio inside exponent becomes nearly zero to make η_L become close to η_L^{max}. However, when all links are lightly loaded ($minC_L^s\gg 0$) and/or slightly unbalanced ($\Delta C_L^s\approx 0$), the exponent goes to zero to make η_L become close to η_L^{min}. The same pattern applies to η_N.

D. Constraints

Equations (3)–(18) formulates general requirements for node and link mapping, as articulated in [2], and the remaining equations are properly defined to characterize additional variables and expressions.

$$\sum_{v^s\in V^s}\xi_{v^r,v^s}=1,\ \forall v^r\in V^r \quad (3)$$
$$\sum_{v^r\in V^r}\xi_{v^r,v^s}\le 1,\ \forall v^s\in V^s \quad (4)$$

Equations (3), (4) ensure that each VN in the VON request is mapped onto a unique SN,

$$\sum_{v^s\in V^s}\xi_{v^r,v^s}\cdot c_{v^s}^s\ge c_{v^r}^r,\ \forall v^r\in V^r \quad (5)$$

Equation (5) ensures that the embedding SN has enough computing capacity to accommodate the VN.

$$\zeta_{e^r,p^s}\le\xi_{s_{e^r}^r,s_{p^s}^s},\ \forall e^r\in E^r,\forall p^s\in P^s \quad (6)$$
$$\zeta_{e^r,p^s}\le\xi_{d_{e^r}^r,d_{p^s}^s},\ \forall e^r\in E^r,\forall p^s\in P^s \quad (7)$$
$$\sum_{p^s\in P^s}\zeta_{e^r,p^s}=1,\ \forall e^r\in E^r \quad (8)$$
$$\sum_{e^r\in E^r}\zeta_{e^r,p^s}\le 1,\ \forall p^s\in P^s \quad (9)$$

Equations (6)–(9) ensure that each VOL is mapped onto a single substrate lightpath, and the end nodes of the lightpath are the SNs that the corresponding VNs are mapped onto,

$$\sigma_{e_1^r,e_2^r}\ge\zeta_{e_1^r,p_1^s}+\zeta_{e_2^r,p_2^s}-1,\ \forall e_1^r,e_2^r\in E^r,\forall p_1^s,p_2^s\in P_{e^s}^s \quad (10)$$
$$\rho_{e_1^r,e_2^r}+\rho_{e_2^r,e_1^r}=1,\ \forall e_1^r,e_2^r\in E^r \quad (11)$$
$$z_{e_2^r}-w_{e_1^r}+1\le B^s\cdot(1+\rho_{e_1^r,e_2^r}-\sigma_{e_1^r,e_2^r}),\ \forall e_1^r,e_2^r\in E^r \quad (12)$$
$$z_{e_1^r}-w_{e_2^r}+1\le B^s\cdot(2-\rho_{e_1^r,e_2^r}-\sigma_{e_1^r,e_2^r}),\ \forall e_1^r,e_2^r\in E^r \quad (13)$$

Equations (10)–(13) together with (15)-(18) ensure that the spectrum assigned to any VOL is contiguous and the spectrum assigned to any two VOLs, whose associated substrate lightpaths have common SFL(s) do not overlap,

$$z_{e^r}-w_{e^r}+1=\sum_{p^s\in P^s}\zeta_{e^r,p^s}\cdot\left\lceil\frac{bw_{e^r}^r}{m_{p^s}^s}\right\rceil,\ \forall e^r\in E^r \quad (14)$$

Equation (14) ensures that the number of FS' assigned to each VOL can just satisfy its BW requirement.

$$\pi_{e^r,e^s}=\sum_{p^s\in P_{e^s}^s}\zeta_{e^r,p^s},\ \forall e^r\in E^r,\forall e^s\in E^s \quad (15)$$
$$\sum_k\delta_{e^r,e^s}^{(k)}=\pi_{e^r,e^s},\ \forall e^r\in E^r,\forall e^s\in E^s \quad (16)$$
$$w_{e^r}\ge w_{e^s,k}^s\cdot\left(\delta_{e^r,e^s}^{(k)}+\pi_{e^r,e^s}-1\right),\ \forall e^r\in E^r,\forall e^s\in E^s,\forall k \quad (17)$$

$$z_{e^r} - z^s_{e^s,k} \le B^s \cdot \left(2 - \delta^{(k)}_{e^r,e^s} - \pi_{e^r,e^s}\right),\ \forall e^r \in E^r, \forall e^s \in E^s, \forall k \tag{18}$$

Equations (15)–(18) ensure that the assigned CSB for each VOL locates in a single MCSB on SFL e^s, and the MCSB's size is not smaller than that of the assigned CSB.

$$\theta'_{e^s,m,n} \le B^s \cdot \theta_{e^s,m,n},\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{19}$$

$$\theta'_{e^s,m,n} \le w'_{e^s,m},\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{20}$$

$$\theta'_{e^s,m,n} \ge w'_{e^s,m} + B^s \cdot (\theta_{e^s,m,n} - 1),\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{21}$$

$$\phi'_{e^s,m,n} \le B^s \cdot \phi_{e^s,m,n},\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{22}$$

$$\phi'_{e^s,m,n} \le z'_{e^s,m},\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{23}$$

$$\phi'_{e^s,m,n} \ge z'_{e^s,m} + B^s \cdot (\phi_{e^s,m,n} - 1),\ \forall e^s \in E^s, 1 \le m,n \le n^s_{e^s} + Y \tag{24}$$

Equations (19)–(21) construct the equality $\theta'_{e^s,m,n} = w'_{e^s,m} \times \theta_{e^s,m,n}$, likewise (22)–(24) construct the equality $\phi'_{e^s,m,n} = z'_{e^s,m} \times \phi_{e^s,m,n}$.

$$\sum_{m=1}^{n^s_{e^s}+Y} \theta_{e^s,m,n} = 1,\ \forall e^s \in E^s, 1 \le n \le n^s_{e^s} + Y \tag{25}$$

$$\sum_{n=1}^{n^s_{e^s}+Y} \theta_{e^s,m,n} \le 1,\ \forall e^s \in E^s, 1 \le m \le n^s_{e^s} + Y \tag{26}$$

$$\sum_{m=1}^{n^s_{e^s}+Y} \phi_{e^s,m,n} = 1,\ \forall e^s \in E^s, 1 \le n \le n^s_{e^s} + Y \tag{27}$$

$$\sum_{n=1}^{n^s_{e^s}+Y} \phi_{e^s,m,n} \le 1,\ \forall e^s \in E^s, 1 \le m \le n^s_{e^s} + Y \tag{28}$$

Equations (25)–(28) together with (29)–(30) ensure one-to-one correspondence between the elements of $w'_{e^s,m}/z'_{e^s,m}$ and $\hat{w}_{e^s,n}/\hat{z}_{e^s,n}$, respectively.

$$\hat{w}_{e^s,n} = \sum_{m=1}^{n^s_{e^s}+Y} \theta'_{e^s,m,n},\ \forall e^s \in E^s, 1 \le n \le n^s_{e^s} + Y \tag{29}$$

$$\hat{z}_{e^s,n} = \sum_{m=1}^{n^s_{e^s}+Y} \phi'_{e^s,m,n},\ \forall e^s \in E^s, 1 \le n \le n^s_{e^s} + Y \tag{30}$$

$$\hat{w}_{e^s,n} \le \hat{w}_{e^s,n-1},\ \forall e^s \in E^s, 2 \le n \le n^s_{e^s} + Y \tag{31}$$

$$\hat{z}_{e^s,n} \le \hat{z}_{e^s,n-1},\ \forall e^s \in E^s, 2 \le n \le n^s_{e^s} + Y \tag{32}$$

Equations (31)–(32) make sure that elements of $\hat{w}_{e^s,n}/\hat{z}_{e^s,n}$ will be sorted in descending order.

$$l_{e^s,m} = \hat{w}_{e^s,m} - \hat{z}_{e^s,m},\ \forall e^s \in E^s, 1 \le m \le n^s_{e^s} + Y \tag{33}$$

$$l_{e^s,m} \le B^s \cdot f_{e^s,m},\ \forall e^s \in E^s, 1 \le m \le n^s_{e^s} + Y \tag{34}$$

$$l_{e^s,m} \ge f_{e^s,m},\ \forall e^s \in E^s, 1 \le m \le n^s_{e^s} + Y \tag{35}$$

Equation (33) constructs pattern of lengths of free slot blocks for every substrate link. Equations (34)–(35) construct $f_{e^s,m}$.

$$\psi_{e^r,e^s} \le B^s \cdot \pi_{e^r,e^s} + w^s_{e^s,1},\ \forall e^s \in E^s, \forall e^r \in E^r \tag{36}$$

$$\psi_{e^r,e^s} \le w_{e^r} + w^s_{e^s,1} \cdot \left(1 - \pi_{e^r,e^s}\right),\ \forall e^s \in E^s, \forall e^r \in E^r \tag{37}$$

$$\psi_{e^r,e^s} \ge w_{e^r} + B^s \cdot \left(\pi_{e^r,e^s} - 1\right),\ \forall e^s \in E^s, \forall e^r \in E^r \tag{38}$$

$$\psi_{e^r,e^s} \ge w^s_{e^s,1} \cdot \left(1 - \pi_{e^r,e^s}\right),\ \forall e^s \in E^s, \forall e^r \in E^r \tag{39}$$

Equations (36)–(39) construct the relations:

$$\psi_{e^r,e^s} = \begin{cases} w_{e^r} & \text{if } \pi_{e^r,e^s} = 1 \\ w^s_{e^s,1} & \text{if } \pi_{e^r,e^s} = 0 \end{cases} \tag{40}$$

$$\Psi_{e^s} \ge \psi_{e^r,e^s},\ \forall e^s \in E^s, \forall e^r \in E^r \tag{41}$$

$$C^{max}_L \ge \sum_{m=1}^{n^s_{e^s}+Y} l_{e^s,m},\ \forall e^s \in E^s \tag{42}$$

$$C^{\min}_L \le \sum_{m=1}^{n^s_{e^s}+Y} l_{e^s,m},\ \forall e^s \in E^s \tag{43}$$

Equations (42)–(43) find the maximum and minimum available link capacities among all SFLs, respectively, after allocating all VOL requests.

$$C^{max}_N \ge c^s_{v^s} - \sum_{v^r \in V^r} \xi_{v^r,v^s} \cdot c^r_{v^r},\ \forall v^s \in V^s \tag{44}$$

$$C^{min}_N \le c^s_{v^s} - \sum_{v^r \in V^r} \xi_{v^r,v^s} \cdot c^r_{v^r},\ \forall v^s \in V^s \tag{45}$$

Equations (44)–(45) find the maximum and minimum available node capacities among all SNs, respectively, after allocating all VON requests.

IV. PERFORMANCE EVALUATION

We implemented the proposed ILP model by ILP solver IBM Cplex [17] to evaluate the blocking performance of a simple six-node topology as substrate network, shown in Figure 1. We assumed that all SFLs have the same length of 100 km, the number of VNs in each VON request is uniformly distributed in a preset range, and the probability that a VN-pair is directly connected equals 0.5, which means that there would be $n(n-1)/4$ VOLs on average for a VON request with n VNs. VONE requests arrive randomly according to Poisson traffic distribution and their holding time follow negative exponential distribution with an average value of 10 min. We also assume that each incoming request should be served instantly or will be rejected forever due to insufficiency of available resources.

 ISBN: 978-1-61208-388-9

The blocking performance of the proposed model under different request loads were then investigated and compared with those of the reference work [2] for three different scenarios through extensive numerical simulations. Table I depicts common parameter settings and Table II illustrates details of scenario-specific parameters. In scenarios 1 and 3, the range of variations of VOL's BW requirement and VN's computing requirement, respectively, were deliberately increased to present the effectiveness of load-balancing and fragmentation-awareness contributions more clearly. To find the optimal values of weighting coefficients, they were initially set to 1 and then each of them was precisely tuned against the others.

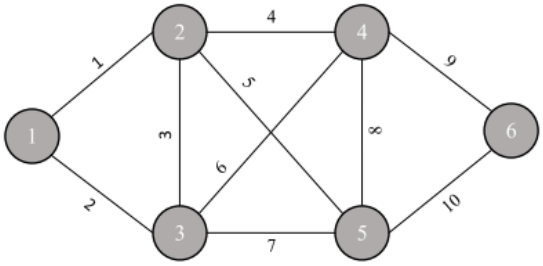


Fig. 1. Six-node topology.

TABLE I. COMMON SIMULATION PARAMETERS

Number of SNs	6
Number of SFLs	10
SN's computing capacity	50 units
SFL's BW capacity	50 FS'
Number of VNs in a VON	[2,3]
Number of VON requests	10,000

TABLE II. SCENARIO-SPECEFIC PARAMETERS

	Scenario1	**Scenario 2**	**Scenario3**
VN's computing requirement (units)	[1,4]	[1,4]	[1,5]
VOL's BW requirement (units)	[1,20]	[1,12]	[1,12]
CSB alignment coeff (α)	1	1	1
Spectral fragmentation coeff (β)	0.5	0.5	0.5
BW utilization coeff (γ)	1	1	1
Link load-balancing coeff (η_L)	[0.5,1.5]	[0.5,1.5]	[0.5,1.5]
Node load-balancing coeff (η_N)	[1,3]	[2,4]	[3,5]

Figure 2 shows the variation of blocking probability versus VON request load for the first scenario under four different situations. The "reference" graph exhibits results obtained by preserving the third term of the objective function and eliminating all other terms, as well as their associated constraints. This served us to benchmark the performance of the proposed model against the previous work. The graph labeled "FA" illustrates results obtained by considering only the impact of Fragmentation-Awareness in VONE ($\alpha = \beta = \gamma = 1, \eta_L = \eta_L = 0$). Conversely, the graph labeled "LB" shows results gained by including only the impact of Load-Balancing VONE ($\alpha = \beta = 0, \gamma = \eta_L = \eta_L = 1$). Finally, the graph labeled "FA+LB" represents that the combined impact of all terms having their optimal weighting coefficients achieves the best performance. This verifies the presumption that all FA and LB terms synergistically contribute to lower the blocking probability.

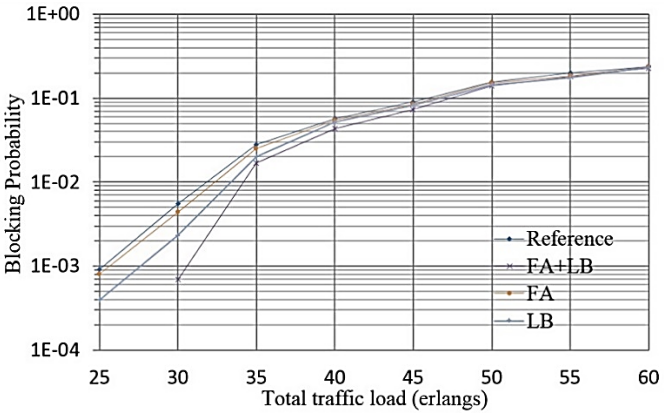


Fig. 2. Scenario 1- $c^r_{v^r} = [1,4]$, $bw^r_{e^r} = [1,20]$.

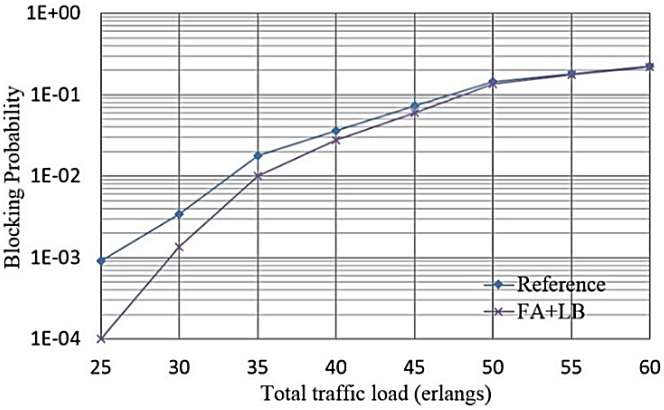


Fig. 3. Scenario 2- $c^r_{v^r} = [1,4]$, $bw^r_{e^r} = [1,12]$.

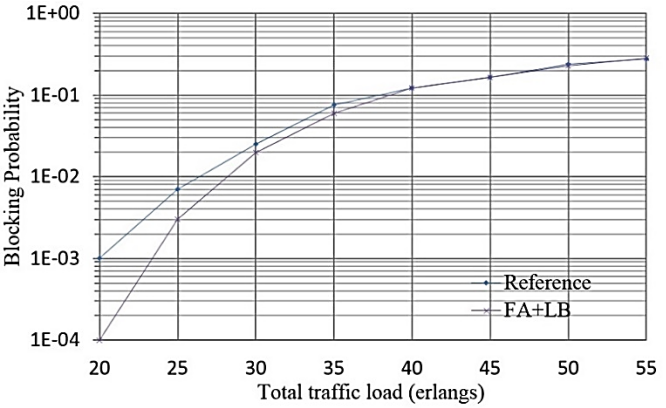


Fig. 4. Scenario 3- $c^r_{v^r} = [1,5]$, $bw^r_{e^r} = [1,12]$.

Variation in the range of VOL's BW requirement or VN's computing requirement, was studied in scenarios 2 and 3, respectively. Figures 3 and 4 show that the proposed scheme could still successfully control spectral fragmentation and simultaneously maintain load-balanced condition. It is worth noting that, under light request load, the proposed scheme fulfilled an order-of-magnitude reduction in the blocking probability in all scenarios compared with the reference work without imposing any additional cost.

 ISBN: 978-1-61208-388-9

V. CONCLUSION AND FUTURE WORK

The recent work on virtual network embedding presented ILP formulation to optimize resource utilization. In this paper, we extended the model so that fragmentation-awareness, as well as load-balancing features were taken in to account in the optimization process. Simulation results verified that our proposed model outperforms the recent work under three different scenarios. Parameter settings in scenario 2 were exactly replicated the reference work. Scenarios 1 and 3 were arranged to demonstrate more clearly the effectiveness of load-balancing and fragmentation-awareness contributions, respectively. Moreover, our model could achieve remarkable reduction in blocking probability under wide range of request traffic loads, particularly in light request traffic loads.

Our future plan is to develop efficient heuristic algorithm based on the notions embodied in ILP formulation.

REFERENCES

[1] J. Socrates-Dantas, et al., "A Study in current dynamic fragmentation-aware RSA algorithms," in Transparent Optical Networks (ICTON), 2014 16th International Conference on, 2014, pp. 1-4.

[2] L. Gong and Z. Zhu, "Virtual Optical Network Embedding (VONE) Over Elastic Optical Networks," Journal of Lightwave Technology, vol. 32, 2014, pp. 450-460.

[3] J. Perello and S. Spadaro, "Virtual network embedding in optical infrastructures," in Transparent Optical Networks (ICTON), 2012 14th International Conference on, 2012, pp. 1-4.

[4] S. Zhang, L. Shi, C. S. Vadrevu, and B. Mukherjee, "Network virtualization over WDM networks," in Advanced Networks and Telecommunication Systems (ANTS), 2011 IEEE 5th International Conference on, 2011, pp. 1-3.

[5] Q. Zhang, et al., "RWA for Network Virtualization in Optical WDM Networks," in National Fiber Optic Engineers Conference, 2013, pp. 1-3.

[6] W. Xie, et al., "Survivable virtual optical network mapping in flexible-grid optical networks," in Computing, Networking and Communications (ICNC), 2014 International Conference on, 2014, pp. 221-225.

[7] J. Zhao, S. Subramaniam, and M. Brandt-Pearce, "Virtual topology mapping in elastic optical networks," in Communications (ICC), 2013 IEEE International Conference on, 2013, pp. 3904-3908.

[8] L. Gong, W. Zhao, Y. Wen, and Z. Zhu, "Dynamic transparent virtual network embedding over elastic optical infrastructures," in Communications (ICC), 2013 IEEE International Conference on, 2013, pp. 3466-3470.

[9] X. Chen, et al., "A novel fragmentation-aware spectrum allocation algorithm in flexible bandwidth optical networks," Optical Switching and Networking, vol. 12, 2014, pp. 14-23.

[10] M. Zhang, W. Lu, Z. Zhu, Y. Yin, and B. Yoo, "Planning and provisioning of elastic O-OFDM networks with fragmentation-aware routing and spectrum assignment (RSA) algorithms," in Asia Communications and Photonics Conference, p. ATh2D. 3, 2012, pp. 1-3.

[11] S. Talebi, et al., "Spectrum management techniques for elastic optical networks: A survey," Optical Switching and Networking, vol. 13, 2014, pp. 34-48.

[12] Y. Yin, Z. Zhu, and S. Yoo, "Fragmentation-aware routing, modulation and spectrum assignment algorithms in elastic optical networks," in Optical Fiber Communication Conference, p. OW3A. 5, 2013, pp. 1-3.

[13] P. M. Moura, N. L. da Fonseca, and R. A. Scaraficci, "Fragmentation aware routing and spectrum assignment algorithm," in Communications (ICC), 2014 IEEE International Conference on, 2014, pp. 1137-1142.

[14] Y. Yin, et al., "Spectral and Spatial 2D Fragmentation-Aware Routing and Spectrum Assignment Algorithms in Elastic Optical Networks [Invited]," Journal of Optical Communications and Networking, vol. 5, 2013, pp. 100-106.

[15] W.-H. Hsu and Y.-P. Shieh, "Virtual network mapping algorithm in the cloud infrastructure," Journal of Network and Computer Applications, 2013, pp. 1-11.

[16] M. Jinno, et al., "Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network [Topics in Optical Communications]," Communications Magazine, IEEE, vol. 48, 2010, pp. 138-145.

[17] IBM-CPLEX. version 12.6. IBM, 2013.

 ISBN: 978-1-61208-388-9

Estimating Working Set Size by Guest OS Performance Counters Means

Anna Melekhova
Parallels
Moscow, Russia
email: annam@parallels.com

Larisa Markeeva
Innopolis University
Kazan, Russia
email: l.markeeva@innopolis.ru

***Abstract*—Cloud infrastructures imply virtual machines dynamic hosting. The distribution of resources is performed in dependence on the volume and a pattern of used resources. An estimation of current load and prediction of future load are mandatory for effective resources management. The statistical processing of internal counters of a guest operating system gives good prospects. The disadvantage of the method is the complexity of data collection and its processing. The large number of parameters and variance in their behavior in different operating systems and configurations introduce extra complexity for the virtualization case. The present research covers the estimation of a virtual machine working set based on guest OS internal counters. The correction of the estimated value is done in accordance on the feedback of donor guest OS.**

Keywords-virtual machine; cloud computing; memory management; working set.

I. INTRODUCTION

Resources virtualization is one of the key topics in IT (Information Technology) industry due to resource optimization task. Most of the nowadays workloads on user resources come in peaks. That is, virtual machines mostly consume little resources; but, eventually, activity gets to a peak and the volume of required resources increases significantly. In this case, resources exhausting would bring a considerable performance loss. Virtualization is a popular solution to handle the issue and make the management more flexible. A widely used practice that increases the percentage of resources utilization is setting the size of virtual resources above the real hardware resources. This method is called overcommit or oversubscription. To make this mode effective a smart resources management is required. This management follows the principle "from each according to his ability, to each according to his needs" when resources are assigned basing on the real consumption level, not the assigned one.

CPU (Central Processing Unit) and memory are the most common overcommitted resources as their utilization is not too high in usual workloads [13][14]. Modern operating systems (OS) behave differently when these resources are not fully utilized. For instance, when OS does not need CPU resource, it sends a halt signal (hlt instruction) to CPU to reduce power consumption. A virtualization system handles this event and thus, it determines that the CPU resource can be diverted to another consumer. At the same time, there is no instruction to signal about unused memory in Intel x86 architecture. Thus, it is impossible to estimate real memory consumption by hardware means.

We define a working set as a set of memory pages used by a consumer (a process or a virtual machine) in a given time frame [1]. The problem of virtual set size estimation is not new; it is discussed in [2]-[6]. There is an analogy with a process in an operating system. A virtual machine does not know that the address space is not continuous, similar to how a process does not know that its allocated space is not continuous. Accesses to unmapped memory are handled by a virtualization engine transparently to the guest operating system, similar to how a page miss is handled transparently to a process. Processes, usually, do not inform operating systems about the required volume of memory. Operating systems do not inform the virtual machine about the required volume of memory as well.

Although a working set topic is elaborately investigated (the fundamental work of Denning dates back to 1970 [7]), the virtualization adds a new dimension. While a process memory is definitely a black box for an operating system, a virtual machine can use a paravirtualization to obtain information on the guest resources management or even improve guest OS for better interaction with the virtualization module. The present article unveils a method that reflects changes inside a guest OS by the means of internal counters statistics. This statistics can be used to predict peak loads as well.

There is a number of techniques to optimize RAM utilization:

- Content-based page sharing
- Ballooning
- Memory compression
- Page replacement algorithms

The detailed description of these techniques can be found at [13][14]. Section II.A of the present article describes ballooning in more details.

Each technique has its advantages and disadvantages, while most virtualization products use a combination of them [13] [14]. A content-based page sharing has an imperceptible performance impact on the host, but its memory utilization ability depends on the workload type. Memory compression decreases the cost of a page miss, but it doesn't influence the number of misses. Page replacement algorithms often don't satisfy an acceptable page miss rate due to so-called semantic gap. Ballooning dramatically decreases the page miss rate but in the case of overinflation it causes the guest OS lags and even falls.

 ISBN: 978-1-61208-388-9

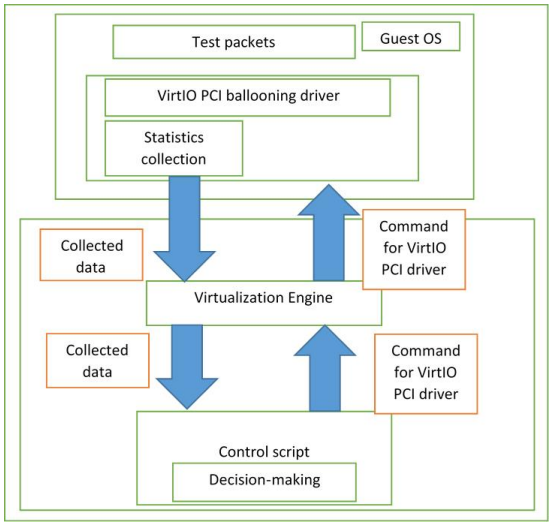


Figure 1. Balloon driver control diagram

The ballooning technique is implemented in all virtualization products – VMWare, Xen, KVM [15], Parallels, Microsoft Hyper-V, Sun VirtualBox Guest ballooning driver is designed for Linux and Windows OSes. But all known balloon implementation has no idea on the best balloon size – they just process external inflate/deflate commands [16].

Luiz Capitulino discusses auto-ballooning in [21]. The proposed concept fits Linux hosts only. The idea is to use three memory pressure types, LOW, MEDIUM, and HIGH and use these states to automatically decide whether to inflate or deflate the balloon.

Another technology used for autoballooning on KVM is Memory Overcommitment Manager (MOM) [22][23]. It also relies on the memory pressure state, same as for the previous technique. In contrast with the previous technique, this uses a set of scripts and the libvirt library [24], thus it does not require kernel modifications. But, MOM works only with Linux guests. A more detailed description of the MOM technique can be found in [23].

Both listed approaches share the same problems: the lack of adequate performance test sets [21] and the non-predictive nature of the techniques [21]-[23]. The latter means that these techniques are unable to forecast future loads and prepare the environment for such loads.

In this paper, we aim to build an effective working set size estimator and use it for an automatic balloon control algorithm. We present performance results for our algorithm as well.

Section II introduces basic definitions such as virtual machine, hypervisor, ballooning and internal counters. Section III describes the architecture of the data collecting subsystem, the data collecting itself, an efficiency test and its configuration. Section IV covers data collection required for the analysis, and introduces the estimation of a working set size. Section V presents performance testing results and methods for performance improvements. Section VI, as a conclusion, lists briefly the obtained results, and proposes further research topics.

II. BASIC DIFINITIONS

Virtual machine (VM) is an emulation of a particular computer system.

A hypervisor is a software, firmware, or hardware that creates and runs virtual machines. We use terms virtualization module, virtualization engine, engine to denote a hypervisor.

A computer that runs a hypervisor, which maintains one or more virtual machines, is called a host machine.

Each virtual machine is called a guest machine.

A. Ballooning

A ballooning is a technique of on-the-fly virtual machine random access memory size modification. Each guest OS gets an additional driver (balloon). The driver is managed by a virtualization engine. The engine issues commands of two types: increase the number of pages allocated by the driver (inflate balloon) or decrease this number (deflate balloon). The memory pages allocated by the driver are committed from the point of view of the guest OS and thus, they could not be allocated to other processes. The hypervisor on the host side gives the memory pages allocated by the balloon driver to other virtual machines running on the same host machine. This implements the principle “from each according to his ability, to each according to his needs”.

B. Operating system internal counters

Further, we present a description of operating system internal counters and how to use these counters.

Operating systems of Microsoft Windows family use system internal counters such as TotalMemory [8] and CommitMemory for computer resources management and statistics collection. The system updates these counters automatically. User space and kernel space programs can fetch these counters at any time with proper requests. The requests to get values of the counters are fast as usually the data is simply copied from the kernel memory to the specified area.

As shown in [9][10], even different versions of Microsoft Windows significantly vary in the policy of resources management. So, the existence of a single efficient resource management policy for the different OSes is unlikely.

This article investigates counters of two OSes within a single family: Windows 7 and Windows 8. This choice is based on the popularity of Microsoft's operating systems family in general [17] and specifically Windows 7 and Windows 8 that still have a mainstream support [18].

The approach illustrated on Figure 1 can be applied to other operating systems such as Linux. However, the differences in system API (Application programming interface), counters, and resource management algorithms [19][20] make modifications to the VirtIO PCI

 ISBN: 978-1-61208-388-9

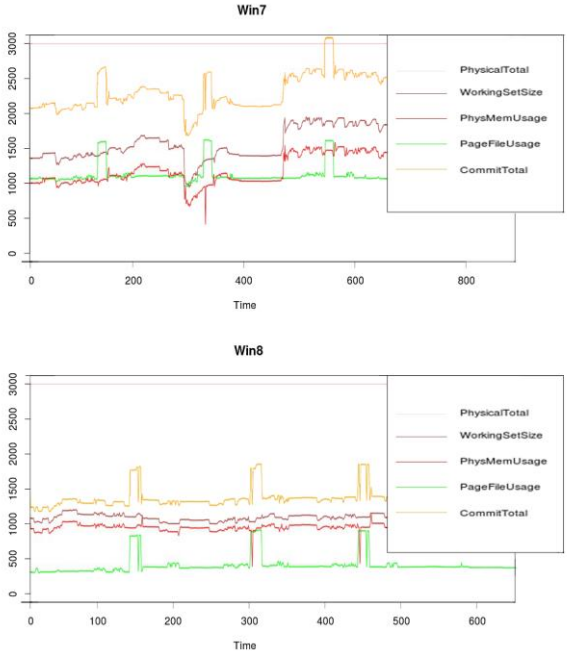


Figure 2. Internal counters dynamics for Windows 7 and Windows 8 operating systems.

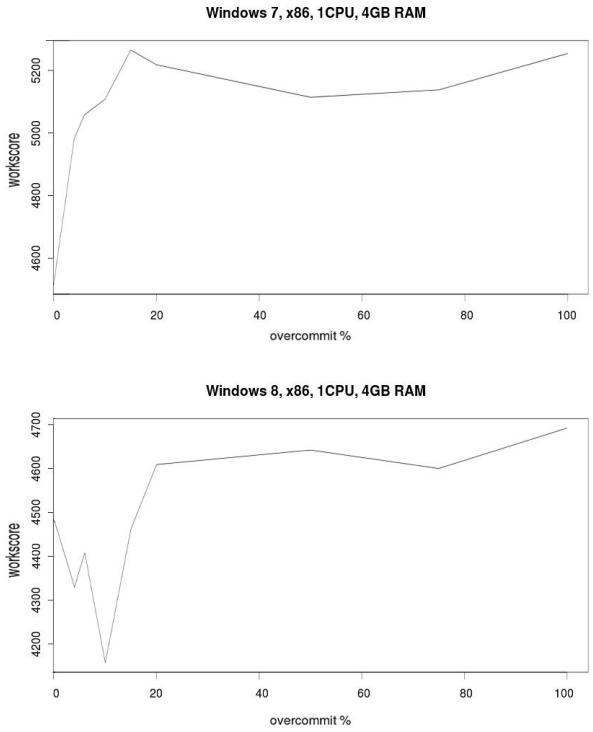


Figure 3. PCMart test results for different σ values in Windows 7 and Windows 8 operating systems.

ballooning driver necessary as well as a search for counters similar to TotalMemory and CommitMemory. The analysis of memory management and swapping algorithms in Linux is another topic for further research.

We deduce formulas (1)-(4) for the estimate of the working set size and test the efficiency of these estimates. The resulting value is applied by balloon means. That is we:

1) Estimate the value with guest OS counters.

2) Set balloon size to take away all potentially unused memory pages.

3) Correct the value in accordance to guest OS counters value.

III. THE DATA COLLECTING

A. An architecture of the data collecting subsystem

In order to minimize the cost of data acquisition, prototyping, and controlling the balloon driver we use the scheme from the Figure 1.

We have modified the VirtIO balloon driver so that in addition to the main goal the driver gets internal counters' values and delivers them to a hypervisor. The hypervisor sends this data to the Control script that analyzes the data and makes decision. Control script can send commands to the VirtIO balloon driver through the hypervisor interfaces.

B. Data collecting and efficiency test

The dependencies between internal counters and the working set size are deduced using special tests that can put virtual machines under specific workloads. These tests should be able to inflict different patterns of workload and measure the overall virtual machine's performance. The main point is that improvements in memory management shouldn't downgrade the performance.

We have used PCMark [11] that gives an aggregated view and atomics that test separate functions of virtual machine. The atomics tests are describe in Table I. The atomics tests measure system performance in a wide variety of workloads: CPU workload, network, HDD, Java programs, 2D and 3D graphics, and so on.

C. Test configuration

Host machine has the following configuration: Intel Core i5-4570 3.2GHz x 4, 16Gb RAM, 1Tb HDD, Intel Haswell Desktop, Linux Ubuntu 13.04.

Virtual machines have the following configuration: 1CPU, 4Gb RAM, 256Mb Video, 64Gb HDD. The operating systems used are Windows 7 x64 and Windows 8 x64.

Virtualization engine used is Parallels Desktop 10.

IV. VIRTUAL MACHINE WORKING SET CALCULATION

A. Collecting data for the analysis

The first modification of the VirtIO balloon driver sent 20 parameters to the hypervisor. During the experiments, some of the parameters turned out to be statistically irrelevant.

 ISBN: 978-1-61208-388-9

TABLE I. PERFORMANCE TESTS

Test	Windows 7, 1CPU, 4Gb	Windows 8, 1CPU, 4Gb
busyloop_test	-2.0%	-2.0%
system_syscall_test	-0.1%	-0.3%
process_exec_test	-2.0%	-2.3%
thread_create_test	-2.0%	-1.8%
io_hdd_seq_rand_rd _test	-99.8%	-99.2%
virtalloc_test	-1.1%	-0.2%
mem_read_test	-1.5%	-5.4%
mem_write_test	-1.6%	-2.9%
mem_pf_read_test	-0.2%	-10.0%
mem_pf_write_test	+0.4%	-9.1%
mem_copy_test	-1.2%	-1.6%

The following parameters remained:

- Physical total - the amount of actual physical memory.
- Commit total - the number of pages currently committed by the system.
- Working set size - current working set size.
- Commit available - the number of pages currently available to the system.
- Physical memory usage - the amount of physical memory currently in use.
- Page file usage - the amount of page file currently in use.
- Balloon size – the memory allocated by the VirtIO balloon driver

Figure 2 demonstrates the collected data for Microsoft Windows 7 and Microsoft Windows 8.

B. *Working set size estimation – the first approximation*

Figure 2 demonstrates that the working set size has a lower bound equal to the Physical memory usage and an upper bound equal to the Commit total.

The lack of random access memory causes memory swapping. Read/write operations become extremely slow, so underestimating the working set size leads to a significant drop in performance of the overall guest operating system. Thus, the Physical memory usage should not be estimated as its lower bound.

We denote the estimate of the working set size by W.

Evidently, the Commit Total value can not be less than the working set size as it includes the volume of used page file pages plus the volume of memory allocated in the RAM. So, it is greater than or equal o the size of a working set.

When the VirtIO balloon driver is disabled, the Commit total contains all the memory pages allocated by the system, hence

$$W=CommitTotal \quad (1)$$

TABLE II. PERFORMANCE RESULTS FOR CACHE SIZES, VARYING FROM 256MB TO 768MB. WINDOWS 7 OS

Test	Windows 7, Compared to a VM with no memory management	Windows 7, Compared to the VM with memory management without cache control
busyloop_test	+1.1%	+3.2%
system_syscall_test	-1.0%	-0.9%
process_exec_test	-3.3%	-1.4%
thread_create_test	+3.1%	+5.0%
io_hdd_seq_rand_rd _test	-99.8%	+32.0%
virtalloc_test	-0.4%	+0.7%
mem_read_test	-2.4%	-0.9%
mem_write_test	-2.4%	-0.8%
mem_pf_read_test	-0.3%	-0.0%
mem_pf_write_test	-0.1%	-0.6%
mem_copy_test	-0.9%	+0.3%

When VirtIO balloon driver is enabled, the Commit total is increased by the size of the balloon. So,

$$W=CommitTotal-BalloonSize \quad (2)$$

C. *Working set size estimation – the second approximation*

The working set size as we calculated it in (2) does not take into account the size of system caches. This raw result would drop the system performance since on disk cache exhausting the number of actual I/O operations increases. Thus, it is reasonable to provide additional memory for OS caches.

We denote the cache memory estimate by O and effective working set size as E, so

$$E=W+O \quad (3)$$

We assume that O depends on the total physical RAM of a virtual machine, hence

$$O=\sigma \cdot PhysicalTotal \quad (4)$$

According to [9][10] policies of memory managements vary, so σ is a constant defined by the operating system version.

The constant σ from (4) is evaluated as follows:

1) Modify the Control Script. In addition to the data collection it calculates the E in accordance to the formula (3).

2) The Control Script sends inflate/deflate balloon commands to the balloon driver via the hypervisor interface. The size is set to *PhysicalTotal-E* where *PhysicalTotal* is the total size of the assigned random access memory.

 ISBN: 978-1-61208-388-9

TABLE III. PERFORMANCE RESULTS FOR CACHE SIZES, VARYING FROM 256MB TO 768MB. WINDOWS 8 OS

Test	Windows 8, Compared to a VM with no memory management	Windows 8, Compared to the VM with memory management without cache control
busyloop_test	-1.6%	+0.4
system_syscall_test	+1.1%	+1.4%
process_exec_test	+2.7%	+5.1%
thread_create_test	+3.1%	+5.0%
io_hdd_seq_rand_rd _test	-98.8%	+53.1%
virtalloc_test	+0.2%	+0.4%
mem_read_test	-0.2%	+5.5%
mem_write_test	+6.1%	+9.2%
mem_pf_read_test	-4.9%	+5.7%
mem_pf_write_test	-4.0%	+5.7%
mem_copy_test	+7.7%	+9.4%

3) The guest machine runs PCMark tests with different values of σ.

A considerable decrease of performance (more than by k percent) can be observed for Windows 7 operating system when $\sigma<0.2$.

In case of Windows 8, the performance drops by more than k percent when $\sigma<0.15$.

The testing results are presented in Figure 3.

V. PERFORMANCE EVALUATION

We used the special tests described in Section III-B to test the performance of our estimation. The results are provided in Table II.

Performance changes significantly in tests: io_hdd_seq_rand_rd_test, for Windows 7 and io_hdd_seq_rand_rd_test, mem_pf_read_test, mem_pf_write_test for Windows 8.

A. Improving VM performance with the required memory size estimates

Operating systems use free memory for file caches, and it is the main cause for the performance downgrade analyzed in Section IV-C. All free memory is occupied by the VirtIO balloon driver, so no memory is left for caches. The following techniques can be used to circumvent this problem:

1) Enable LargeCacheFile to change the cache control policy [12].

2) Set the limits on the file cache using system calls.

The file cache size must be chosen individually, depending on user's needs. Tables III and IV contain performance results for cache sizes varying from 256Mb to 768Mb.

A considerable increase of performance in the case of controlled cache file size opposed to the uncontrolled cache size with the same estimate is obvious. Still, a significant drop of performance compared to a virtual machine with no memory management persists. But the resource gain on the system is significant – about 25% of memory in case of Windows 7 and about 30% of memory in case of Windows 8 were retained, assuming that the declared memory size in a VM was 4096Mb.

VI. CONCLUSION AND FUTURE WORK

We have obtained estimates on the working set size of virtual machines. These estimates were modified to include cache size in order to gain back the performance. Such an approach gave an acceptable performance while achieving the significant resource economy. Unfortunately, the current approach considers recalculation of the value for each operating system.

We propose a possible solution of this problem. One can collect lots of statistical data for different guest operating systems and build a predicting system such as an autoregressive model and/or a neural network. These systems are promising in terms of forecasting operating system working sets. Additional research in this area is required to make this idea practical, including the analysis of statistical properties of the memory utilization and internal counters time series.

As further research topics we propose improvements to the obtained estimate and a more elaborate search for dependencies between the working set size and internal counters.

ACKNOWLEDGMENT

This work has been supported by the Russian Ministry of education and science with the project "Development of new generation of cloud technologies of storage and data control with the integrated security system and the guaranteed level of access and fault tolerance" (agreement: 14.612.21.0001, ID: RFMEFI61214X0001).

REFERENCES

[1] A. Tanenbaum, Modern Operating Systems Third Edition, pp. 209 – 210, 2009.

[2] P. J. Denning, “The working set model for program behavior”, Commun. ACM 11, 5 May 1968.

[3] P. J. Denning, “Working Sets Past and Present”, IEEE Trans. Softw. Eng. 6, 1, 1980, pp. 64-84,.

[4] Weiming Zhao, Xinxin Jin, Zhenlin Wang, Xiaolin Wang, Yingwei Luo, and Xiaoming Li, “Low cost working set size tracking”. In Proceedings of the 2011 USENIX conference on USENIX annual technical conference (USENIXATC’11). USENIX Association, Berkeley, CA, USA, 2011, pp. 17-17,.

[5] Working set, [retrieved: January, 2015], from http://msdn.microsoft.com/enus/library/windows/desktop/cc441804(v=vs.85).aspx.

[6] D. R. Slutz, I. L. Traiger, “A note on the calculation of average working set size”. Commun. ACM 17, 10, 1974, pp. 563-565.

[7] P. J. Denning, “Virtual Memory”, ACM Comput. Surv. 2, 3, 1970, pp. 153-189.

 ISBN: 978-1-61208-388-9

[8] Performance_information structure, [retrieved: January, 2015], from http://msdn.microsoft.com/en-us/library/windows/desktop/ms684824(v=vs.85).aspx.

[9] L. B. Markeeva, A. L. Melekhova, A. G. Tormasov, "Odnorodnost' virtualizacionnih sobytij, porojdennyh razlichnymi operacionnymi sistemami", Trudi MFTI, vol. 6, N3(23), 2014, pp. 57-64 [in Russian].

[10] S. Sinofsky, Reducing runtime memory in Windows. [retrieved: January, 2015], from http://blogs.msdn.com/b/b8/archive/2011/10/07/reducing-runtime-memory-in-windows-8.aspx.

[11] PCMark. [retrieved: January, 2015], from http://www.futuremark.com/benchmarks/pcmark8.

[12] M. Friedman, O. Pentakalos, Windows 2000 Performance Guide, pp. 280-293, 2002.

[13] S. D. Lowe, Best Practices for Oversubscription of CPU, Memory and Storage in vSphere Virtual Environments, pp. 3-10.

[14] A. Melekhova, "Machine Learning in Virtualization: Estimate a Virtual Machine's Working Set Size", Cloud Computing (CLOUD), 2013 IEEE Sixth International Conference, June 28 2013-July 03 2013.

[15] Main Page - KVM, [retrieved: January, 2015], from http://www.linux-kvm.org/page/Main_Page.

[16] Projects/auto-ballooning – KVM, [retrieved: January, 2015], from http://www.linux-kvm.org/page/Projects/auto-ballooning.

[17] Usage share of operating systems, [retrieved: January, 2015], from http://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0.

[18] Windows lifecycle fact sheet, [retrieved: January, 2015], from http://windows.microsoft.com/en-us/windows/lifecycle.

[19] R. Love, Linux Kernel Development Third Edition, pp. 231-260, 2010.

[20] M. E. Russinovich, D. A. Solomon, and A. Ionescu, Windows Internals, Part 2: Covering Windows Server® 2008 R2 and Windows 7, Sixth Edition, pp. 233-402, 2012.

[21] L. Capitulino, Automatic ballooning, [retrieved: January, 2015], from http://www.linux-kvm.org/wiki/images/f/f6/Automatic-ballooning-slides.pdf.

[22] Memory Overcommitment Manager, [retrieved: January, 2015], from https://github.com/aglitke/mom.

[23] A. Litke, Automatic Memory Ballooning with MOM, [retrieved: January, 2015], from https://aglitke.wordpress.com/2011/03/03/automatic-memory-ballooning-with-mom/

[24] Libvirt virtualizetion API, [retrived: January, 2015], from http://libvirt.org

T-KVM: A Trusted architecture for KVM ARM v7 and v8 Virtual Machines Securing Virtual Machines by means of KVM, TrustZone, TEE and SELinux

Michele Paolino, Alvise Rigo, Alexander Spyridakis, Jérémy Fanguède, Petar Lalov and Daniel Raho
Virtual Open Systems
Grenoble, France
Email: {m.paolino, a.rigo, a.spyridakis, j.fanguede, p.lalov, s.raho} @virtualopensystems.com

***Abstract*—The first market release of Advanced RISC Machines (ARM) v8 System on Chips (SoCs) has created big expectations from smart devices, servers and network equipment vendors, who see compelling advantages in integrating them into their systems. As a consequence software stack deployments for ARMv8 platforms translate market requirements to support OpenStack, Network Functions Virtualization (NFV), Mobile Edge Computing (MEC), In-Vehicle Infotainment (IVI) automotive functions. At the same time, ARMv8 will empower Internet of Things (IoT), Cyber Physical Systems (CPS) and user convergence devices. In this context, virtualization is a key feature to enable the cloud delivery model, to implement multitenancy, to isolate different execution environments and to improve hardware/software standardization and consolidation. Since guaranteeing a strict ownership of both the data and the code executed in Virtual Machines (VMs), which belong to governments, companies, telecom operators and private users, counts more than ever, the security of the hypervisor and its guests has become dramatically important. In this paper, Trusted Kernel-based Virtual Machine (T-KVM), a novel security architecture for the KVM-on-ARM hypervisor, is proposed to satisfy the current market trend. T-KVM integrates software and hardware components to secure guest Operating Systems (OSes) and enable Trusted Computing in ARM virtual machines. The proposed architecture combines four isolation layers: ARM Virtualization and Security Extensions (also known as ARM VE and TrustZone), GlobalPlatform Trusted Execution Environment (TEE) APIs and SELinux Mandatory Access Control (MAC) security policy. The T-KVM architecture can be implemented on platforms based on ARM v7 and v8 architectures, without requiring additional custom hardware extensions, since, starting from Cortex-A15 (ARM v7 architecture) released in 2012, both the ARM VE and TrustZone are made available. In this paper the T-KVM architecture is described in details, as well as its key implementation challenges and system security considerations. Lastly, a performance evaluation of the proposed solution is presented.**

Keywords–Trusted KVM, KVM Security, ARMv8 Trusted Computing, KVM TrustZone, ARM Virtualization, SELinux, TEE, ARM VMs Security.

I. INTRODUCTION

The use of virtualization in ARM platforms is rapidly increasing due to the deployment of SoCs based on this architecture in different environments such as: servers, Cloud and High Performance Computing (HPC), NFV, MEC, IoT, CPS, smart devices, etc. This technology enables multiple OSes to run unmodified on the same hardware, thus sharing system's resources such as memory, CPUs, disks and other devices. These resources are frequently target of specific virtualized environment attacks (e.g., CPU cache [1], memory bus [2] and VM's devices [3] [4]). For this reason, the security of the virtualized systems is critical. Historically, isolation has been used to enhance the security of these systems [5], because it reduces the propagation risks in compromised environments [6].

The aim of this paper is to propose T-KVM, a novel security architecture for virtualized systems based on four isolation layers: KVM, ARM TrustZone, GlobalPlatform TEE API and SELinux. The former is considered the most popular hypervisor deployed in OpenStack [7], which is a key solution for Cloud, NFV and HPC computing. KVM for ARM is part of the Linux kernel starting from the version 3.9; it is the Linux component that, exploiting the ARM Virtualization Extension, allows to create a fully-featured virtualization environment providing hardware isolation for CPU, memory, interrupts and timers [8]. TrustZone is an hardware security extension for ARM processors and Advanced Microcontroller Bus Architecture (AMBA) devices [9] designed to drastically improve security inside the ARM ecosystem. The extension starts from the assumption that a system, in order to deliver secure services, has to decouple the resources used for general purpose applications from those that handle security assets. To this end, TrustZone creates two hardware isolated partitions in the system: the Secure and the Non Secure World. While the Non Secure World runs a standard OS with optionally a hypervisor, the Secure World contains, handles and protects all the sensitive data of the system. These two worlds are linked together through the GlobalPlatform TEE API [10] [11], a set of specifications for a secure Remote Procedure Call (RPC) mechanism between the trusted and non-trusted compartment of the system. At the time of writing, these specifications don't support virtualization, preventing the use of TPM services inside virtual machines. This work addresses this limitation proposing a design of a set of virtualization extensions to enable the guest operating systems to make use of TPM services provided by the TrustZone Secure World. The latter T-KVM isolation layer is Security-Enhanced Linux (SELinux), a Mandatory Access Control (MAC) solution which brings type enforcement, role-based access control and Multi-Level Security (MLS) to the Linux kernel [12]. By means of these, SELinux confines processes in security domains, where the interaction with other processes and files is permitted only if there is a specific SELinux policy rule which allows it.

In T-KVM, the above technologies are combined and adapted to work together, providing a high security level to the guest applications, without the need of specific hardware or software. As a matter of fact, the proposed architecture relies on open source (KVM and SELinux) components, public specifications (GlobalPlatform TEE Internal and Client APIs) and available hardware features (ARM TrustZone and VE). For these reasons, T-KVM can be easily ported to currently available ARM platforms and Cloud Infrastructure systems

 ISBN: 978-1-61208-388-9

such as OpenStack.

The remaining part of this paper is organized as follows: Section II provides more details about the main security components of the proposed architecture. Section III contains details about the T-KVM architecture, its implementation and security considerations while Section IV present a performance analysis of the overhead introduced by the proposed solution. The related work is presented in Section V and Section VI concludes the paper.

II. The Security Components

In this section, the isolation layers which characterize the T-KVM architecture are described.

A. KVM hypervisor

A hypervisor is a software layer which is able to create virtual instances of hardware resources such as CPUs, memory, devices, etc. in order to enable the execution of multiple operating systems on the same hardware. Different implementation approaches lead to different hypervisor types: a type 1 hypervisor, is a bare metal hypervisor which runs directly on the hardware (XEN or VMWare ESX). A type 2 hypervisor is, on the other hand, a hypervisor which runs inside an operating system (Oracle VirtualBox or VMWare Workstation) at the application layer. Usually, the latter is used in less critical applications [13] because of its dependency from the underlying operating system.

KVM is a hypervisor included in the Linux kernel and available for ARM, x86 and s390 architectures. It is neither a type 2 hypervisor because it does not run as a normal program inside Linux, nor is a typical type 1 hypervisor, because it relies on the Linux kernel infrastructure to run. KVM exploits the CPU Virtualization Extensions to execute guest's instructions directly on the host processor and to provide VMs with an execution environment almost identical to the real hardware. Each guest is run in a different instance of this execution environment, thus isolating the guest operating system. For this reason, this isolation has been used for security purposes [14] [15] [16] [17] in many scientific works. In the ARM architecture, the KVM isolation involves CPU, Memory, Interrupts and timers [8].

B. TrustZone

ARM TrustZone is a set of hardware security extensions for ARM processors and AMBA devices. With TrustZone, the hardware platform is split in two parts, the Secure and the Non Secure Worlds. In order to isolate these two compartments, TrustZone requires: CPU with ARM Security Extensions (SE) along with TrustZone compliant MMU, AMBA system bus, interrupt and cache controllers. Hence the isolation provided by TrustZone includes CPU, AMBA devices, interrupts, memory and caches.

The Secure World is considered trusted, and is responsible for the boot and the configuration of the entire system. In fact, the CPU has banked registers for each World, and security specific configurations can be performed in Secure World mode only. This compartment contains the root of trust of the system and protects sensitive data. The access to AMBA peripherals such as fingerprint readers, cryptoghrapic engines, etc. can be restricted only to the Secure World, thus protecting security devices.

On the other hand, the Non Secure World is intended to be the user's World. In this untrusted compartment, a standard operating system (i.e., Android or Linux) is run. Secure operations such as the access to a secret or the execution of a security algorithm are provided to the user's application running in this compartment by the services run in the Secure World.

These two compartments interact with each other through a specific CPU mode, namely the Monitor Mode. It typically runs a secure context switch routine and is capable of routing interrupts, depending on the configuration, either to the Secure or Non Secure World.

Moreover, the use of the ARM VE Extensions, and consequently of KVM, is possible only in the Non Secure World.

ARM TrustZone is compliant with the GlobalPlatform TEE System Architecture specification [18], which defines the attributes that the hardware must have to properly execute a TEE.

C. GlobalPlatform TEE

GlobalPlatform defines the TEE as an execution environment, which provides security features such as isolated execution, integrity of Trusted Applications (i.e., applications run in the TEE) along with confidentiality of their assets [18]. This isolation protects Trusted Applications (TA) and their data from the Rich Execution Environment (REE), the environment where a standard operating system such as Linux or Android is run. Figure 1 depicts the standard architecture of a GlobalPlatform TEE compliant system.

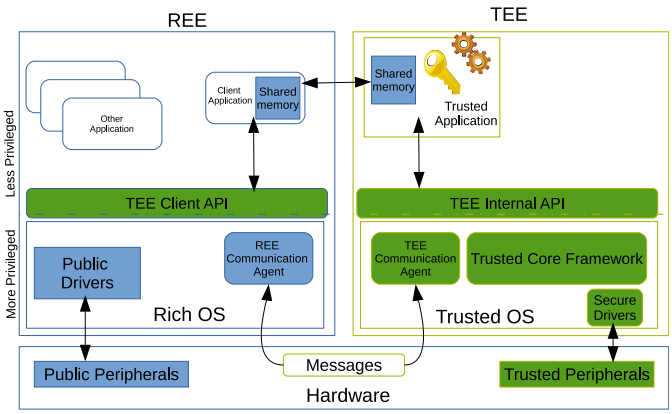


Figure 1. Standard TEE architecture

In order to isolate the TEE from the REE, GlobalPlatform provides a set of specifications which include the following software components [10] [11]:

- The Trusted Core Framework is a common abstraction layer which provides to the TEE Internal API OS-like functions, such as memory management, entry points for TAs, panic and cancellation handling, TA properties access, etc.
- The TEE Client API is an Inter Process Communication (IPC) API that deals with the communication between the REE and the TEE. It allows the applications in the REE (Client Applications or CAs) to leverage the services offered by the TEE.

 ISBN: 978-1-61208-388-9

- The (TEE and REE) communication agents provide support for messaging between the CAs and the TEE. They interact with the Monitor mode to request a context switch between the two Worlds.
- The TEE Internal API allows the TAs to leverage the services offered by the TEE through the following APIs: Trusted Storage for Data and Keys, TEE Cryptographic Operations, Time, and TEE Arithmetical.

Lastly, it is worth to mention that the deployment of a GlobalPlatform compliant solution enables the use of existing TA and CA applications. This is a very important factor, especially in an environment such as the embedded trusted computing, where by tradition the security solutions were developed each time from scratch to address new device families.

D. SELinux

SELinux is a software implementation of the MAC security policy available in the Linux kernel as Linux Security Module (LSM). The key feature of MAC is that the access control decisions are not at discretion of individual users, root included [12]. Thus, once the system security policies have been defined and loaded at boot time in the kernel, they can not be modified. In this way, the subject (e.g., a process) access to objects (e.g., file, socket, etc.) is enforced in the system.

The very same concept can be applied to virtual machines using sVirt, which is a feature of the libvirt library. sVirt installs a set of virtualization specific security policies and automatically tags VMs and their resources in order to isolate guest systems. This isolation prevent any access to VM's resources (disk/kernel files, shared memory, etc.) from external subjects (other VMs, the root user, etc.).

For this and for performance reasons [19], the use of SELinux in virtualized systems is encouraged.

III. The Trusted Hypervisor: T-KVM

T-KVM is a secure hypervisor architecture based on KVM, which combines a Trusted Computing solution such as TrustZone with GlobalPlatform TEE and SELinux. In Figure 2, all the components described in Section II are shown together, composing the T-KVM architecture.

In T-KVM, the GlobalPlatform TEE and REE are respectively implemented inside the TrustZone Secure and Non Secure Worlds. For this reason in the remaining part of this paper, Secure World/TEE and Non Secure World/REE are used as synonyms. These two hardware-isolated environments are linked together with a virtualization-enabled implementation of the GlobalPlatform TEE specifications. The virtualization provided by KVM further isolates the user's applications, enabling the use of different operating systems. This eases multitenancy in server and Cloud environments, and enables Bring Your Own Device (BYOD) [20] paradigm in smart devices. In addition, SELinux isolates in software the virtual machines, protecting guests and their resources from the other virtual machines and the host itself (e.g., malicious cloud administrators, host privilege escalation exploits, etc.). Lastly libvirt, the main virtualization API used by OpenStack to interact with KVM, takes automatically care of the policy configuration and the tag assignment through its component sVirt.

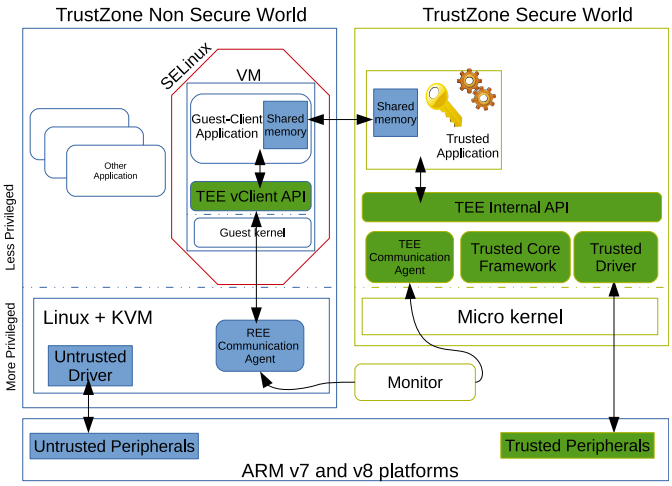


Figure 2. T-KVM architecture, which includes KVM, TrustZone, SELinux and virtualized TEE

A. Implementation details

The following part of this manuscript lists the T-KVM implementation challenges and proposes viable solutions.

1) Trusted boot: The first step of the system's chain of trust is performed during the boot procedure. In fact when the machine boots, the security configuration of the system is not yet in place, and as a result the system is vulnerable to attacks which target to replace the boot procedure.

In order to minimize this risk, the T-KVM's first stage bootloader is a tiny program stored in a on-chip ROM along with the public key needed for the attestation of the second stage bootloader. Since the first stage bootloader is stored in a read-only memory, it can not be updated and it's therefore critical for the security of the system. Soon after the initialization of the key system components, it checks the integrity and boots the second stage bootloader which is located in an external non-volatile memory (e.g., flash memory). The second stage bootloader then loads the microkernel binary in the system's Secure World memory and boots it.

The third stage bootloader of the T-KVM Trusted boot mechanism is a Trusted Application inside the TEE. In fact, when the Secure World OS is up and running, a specific TA checks the integrity of the Non Secure World OS binary (i.e the Linux kernel) and its bootloader (fourth stage). If this last security check is successful, the fourth stage bootloader runs the Non Secure OS and the system can be considered running. On the other hand, if only one of these checks fails, the boot process will be stopped and the machine will enter in a secure and not operational state. Figure 3 shows the T-KVM Trusted boot chain of Trust.

The Trusted boot process is the key element for the attestation of the user space applications because it ensures the integrity of the chain of trust. T-KVM runs in the TEE an attestation service, which is able to check at any moment the integrity of its key components i.e., Quick Emulator (QEMU) [21], libvirt, the VMs and their resources, etc. libvirt in particular, is extended to attest the VMs identity and integrity at each boot and in an event-driven manner, assuring to users and cloud administrators/providers the authenticity

 ISBN: 978-1-61208-388-9

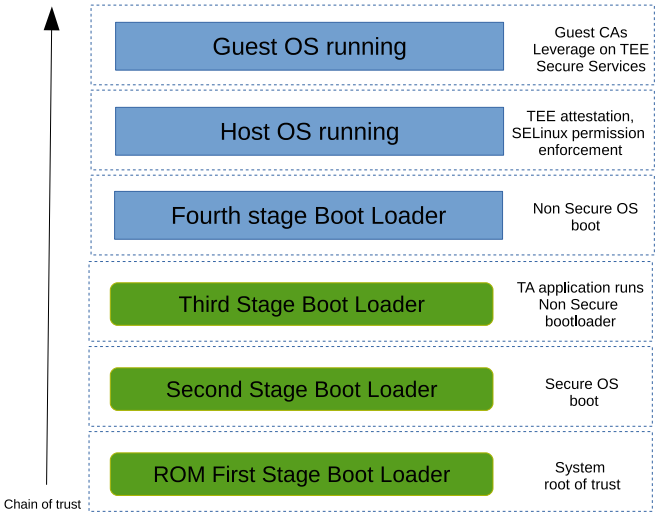


Figure 3. T-KVM Trusted boot procedure

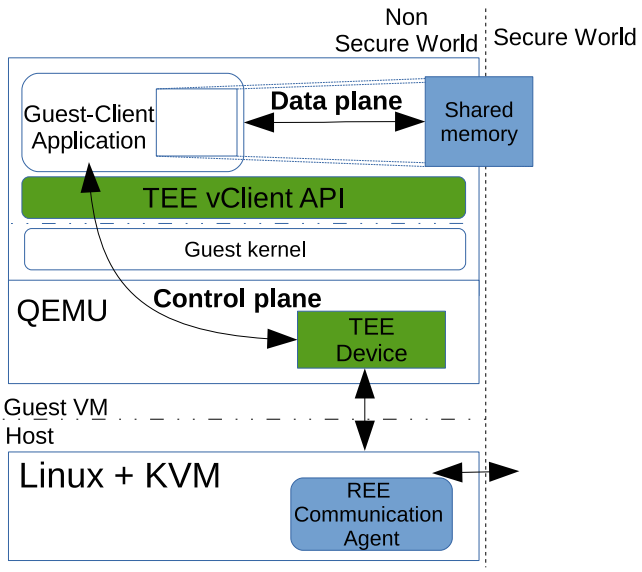


Figure 4. TEE support for Virtual Machines in T-KVM

of the workloads run on the hardware. The security assets (fingerprints, keys, etc.) of these binaries are stored in the Secure World and by consequence can be updated frequently.

2) GlobalPlatform TEE support for virtualization: The main novelty introduced by T-KVM is the support for Trusted Computing inside the Virtual Machines. The virtualization of the TEE functions is of utmost importance for the T-KVM architecture, because it links together the applications run in the VMs with the secure services available in the TrustZone Secure World. At the time of writing, the use of the TEE functions in guest operating systems is not included in the GlobalPlatform API Specification. To enable this feature, T-KVM virtualizes the GlobalPlatform TEE APIs, executing the TEE Client API directly in the Guest Operating System. In order to be as much as possible compliant with the GlobalPlatform Specification and to be able to run CAs also at the host level, T-KVM TEE Client API is the only virtualization aware component.

This awareness needs support from the hypervisor infrastructure. For this reason, as depicted in Figure 4, a specific QEMU device is used to implement the TEE control plane and set up its data plane. All the requests (e.g., initialization/close session, invoke command, etc.) and notification of response are sent to the TEE Device, which delivers them either to the TAs or to the CAs running on the guest OS. To provide good data throughput and latency performance, the data plane is based on a shared memory mechanism. Thus when a response notification arrives from the TrustZone Secure World, the TEE device notifies with an interrupt its driver, which forwards the related information to the Guest-Client Application. The Guest-CA is now able to read the data from the shared memory, without involving the TEE device in the data transfer.

3) Shared memory: T-KVM needs a zero copy shared memory mechanism to share data between the two TrustZone Worlds and between the virtual machine and the host. The latter in particular is very important in systems where VMs need to communicate with each other frequently e.g., NFV, HPC, MEC, etc. Host-guest shared memory mechanisms which provide high performance and low latency already exist for the KVM hypervisor [22] [23]. What these mechanisms actually lack is the support for TrustZone.

By design, the TrustZone Secure World is able to access the full Non Secure World address space. For this reason, the Trusted Applications are able to read/write the content of the VMs shared memory unless they know the address where the shared memory area begins. In order to pass this information, the TEE device control plane extends the T-KVM shared memory mechanism, enabling it to send the shared memory address to the Secure World applications. This mechanism needs to be secured, especially in the Non Secure World, to prevent attacks and information leakage. T-KVM relies on SELinux to define specific access rules for shared memory and to enforce the shared memory access only to the interested parties.

The encryption of the shared memory area is mindfully not considered because, unless hardware accelerators are present in the platform, there would be a performance loss.

4) Secure World: One of the most important parts of the T-KVM architecture is the software running in the TrustZone Secure World. The operating system running in the Secure World should be fast, secure, ideally real-time and free of programming errors (implementation correctness).

The T-KVM architecture empowers the Secure World environment with a microkernel. The primary motivation behind microkernels is the small code footprint, which lead to a smaller attack surface and an easier process of formal verification of the code. In this context, a good candidate is for example seL4, an opensource third-generation microkernel based on L4 and formally verified for functional correctness [24].

The microkernel will run in its userspace the implementation of the GlobalPlatform APIs, the secure device drivers and the TAs. In order to do this, the Secure World OS does not use the main platform storage device to store files and the security assets of the system. An external, non-volatile memory configured by the Secure World as not accessible by the Non Secure World, is used to this purpose.

Finally, a possible alternative to microkernels is a secure

 ISBN: 978-1-61208-388-9

library running in the Secure World such as OPTEE [25]. Despite this solution has a code footprint even smaller than a microkernel, T-KVM uses a microkernel because of its real time features and a higher flexibility for TA developers.

B. Security considerations

Virtual Machines are widely used because of their flexibility and capability to run any operating system. Nonetheless, in order to achieve a higher security level of the system, it is suggested to run single-application operating systems in the T-KVM virtual machines. An example of such an operating system is OSv [26], an opensource solution which is going to be ported to the ARM v8 architecture. For this reason, this work does not discuss the security of the applications inside the virtual machines.

The threat model considered in this paper allows the attacker to completely control one or more virtual machines, both at user and kernel space level. In addition, the cloud administrator, who is permitted to remotely control the virtualized system, is considered as a potential attacker for the identity and integrity of the data.

The security of T-KVM is based on two main assumptions: the attacker does not have physical access to the virtualized system and the first stage bootloader is flawless (thus, the chain of trust is not compromised).

T-KVM has been designed to be compliant with additional hardware accelerators and security modules. For example, SecBus [27] can be used to protect the system against physical attacks on the memory components (e.g., cold boot), Direct Memory Access (DMA) attacks and on-board probing of the external memory bus. Solutions like NoC Firewall [28], can enhance the compartment isolation granularity at VM level, protecting the system against logical attacks (virus, Trojans) or security vulnerabilities, e.g., corrupt DMA engines.

IV. Experimental Results

The different isolation layers which compose T-KVM provide high security, but at a cost of additional overhead. As a matter of fact, a request for a security service from a virtual machine has to pass through the TEE, the host system and SELinux to arrive in the TrustZone Secure World.

For the T-KVM performance analysis of this paper, we focused on the hardware isolation provided by the hypervisor and TrustZone, as the SELinux performance has been measured by the authors in the past [19], and the TEE overhead will be detailed in future works.

For this reason, following the path of a Secure Monitor Call (SMC) from the guest to the Secure World (and then back in the guest) we measured the overhead introduced by T-KVM.

SMC has been added to the ARM instruction set by the ARM Security Extension and it is used to request the execution of a software routine in the TrustZone Secure World passing through the TrustZone Monitor. In the standard KVM implementation, when the SMC instruction is run by a guest OS, its execution is trapped by KVM, which injects an undefined instruction in the guest, forcing it to handle this accordingly. In T-KVM instead, when such instruction is run, the hypervisor traps the guest SMC execution, modifies its arguments and forwards them to the TrustZone Secure World.

In this scenario, two SMC context switches are involved: firstly from Guest to Host, then from Non Secure to Secure World Mode. The overhead assessment of these two context switch operations is the target of the following analysis.

For the first measurement, we implement a bare metal binary blob for KVM which initializes the Performance Monitoring Unit (PMU) and executes the SMC instruction in the guest. The SMC is then trapped by KVM, which has been modified to immediately return the control to the VM. As soon as the program flow returns back to the guest, it checks the PMU cycle counter status and calculates the overhead. In this way, we are able to measure the overhead of a round-trip context switch between the Guest and the Host when an SMC call is executed.

On the other hand, for the measurements of the context switch overhead between the Non Secure and the Secure Worlds, the PMU cycle counter is set by a Linux kernel module in the host, which executes soon after the world switch request (i.e., the SMC instruction). This provokes an additional switch to the TrustZone Monitor, which saves the Normal World registers, loads the Secure World status and finally jumps to the Secure World. In order to measure the T-KVM context switch cost and not add further overhead, the Secure World immediately runs the SMC call, without executing any meaningful operation. When this instruction is executed in the Secure World, it provokes again a switch to the Monitor Mode, which will now save the secure world context, restore the non secure context, and jump back to the Non Secure World. As soon as the context switch is completed, the Linux kernel reads the PMU cycle counter state and computes the overhead.

The results of both of the above measurements are defined minimal because they are not considering any additional work performed at the destination where they are trapping to. However, in a real scenario a trap is followed by the execution of emulation code for a Guest-Host trap, or some secure service for a Non Secure-Secure trap.

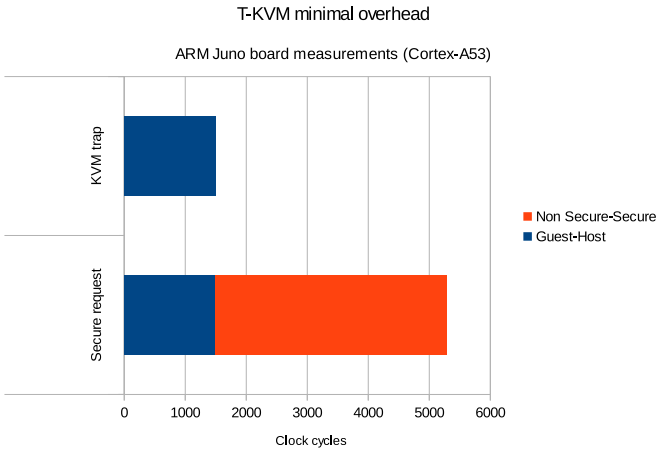


Figure 5. T-KVM overhead measurements

The performance overhead analysis has been performed on an ARMv8 Juno board, which is equipped with two Cortex-A57 and four Cortex-A53 in big.LITTLE configuration. Each test has been repeated five hundred times, running Linux

 ISBN: 978-1-61208-388-9

version 3.17 in both the host and the guest environments. ARM Trusted Firmware [29] has been used as firmware instrastructure. All the workloads for the measurements have been executed on the Cortex-A53, which is the default CPU that the platform uses to execute the Secure World OS.

Figure 5 shows that for a Secure request, which is the sum of the the two measurements described before in this section, the cost is in average 5200 clock cycles. This value represents the minimum overhead that a guest system has to pay to request a secure service to the T-KVM Secure World (the Guest-Host minimum overhead is about 1400, while the Non Secure-Secure is about 3700). This has been compared with the minimum overhead that KVM spends to trap the SMC instruction and perform a context switch between the guest and the host, which is what has been described above in the Guest-Host context switch measurement description. This result (in average about 1400 clock cycles) has been measured with the SMC instruction, but it is valid for all the instructions trapped in KVM, as we did not add any specific code to trap the SMC instruction to the standard KVM implementation.

Finally, it is important to notice that the Non Secure-Secure context switch is 2.5 times slower than the Guest-Host. The main reason for this behaviour is in the number of instructions needed to complete the two operations. In particular, the number of registers that the system has to save and restore for the Guest-Host context switch is significantly lower.

V. Related Work

The hypervisor security is a controversial topic in literature. As a matter of fact, solutions like NoHype [30] [31] propose to secure the virtual machines removing the hypervisor, while others use the hypervisor isolation for security applications [15] [17]. In other scientific works, when compared with TrustZone as a security solution, the hypervisor proves a better flexibility e.g., the Secure World is not able to interpose on and monitor all the important events in the Non-Secure World [16]. The proposed architecture considers the hypervisor as an additional isolation layer, while protecting the security assets through the ARM Security Extensions (TrustZone). T-KVM, combining both solutions and relying on the attestation enabled by the secure boot's chain of trust, is able to provide monitoring features and high security.

In fact, attestation and integrity checks are of paramount importance for the security systems because they allow system designers and administrators to consider a software component as trusted. SecVisor [32], HyperSentry [33] and SPROBES [34] propose different solutions designed for this purpose: the first checks the integrity of commodity OS kernels running the attestation code in hypervisor mode, thus not addressing virtualization. The second enables integrity measurement of a running hypervisor (i.e., XEN) through Intel TxT, hence targeting the x86 architecture. The latter uses TrustZone to enforce kernel code integrity, but without mentioning the attestation challenges in virtualized systems.

These systems are explicitly addressed by solutions such as vTPM [35] or sHype [36], both focusing their efforts on Intel architectures. vTPM proposes a mechanism for the virtualization of TPM functions which dedicates a VM to route and manage the TPM requests, while sHype integrates the MAC security policy directly inside a typical type 1 hypervisor (i.e., XEN).

On the other hand, the solutions proposed by Narari [37] and Lengyel [38] are designed for ARM devices with virtualization extensions. The first proposes a security architecture with TrustZone, SELinux and virtualization, targeting resource constrained devices. The second combines a hypervisor (XEN) and MAC Security policies (XEN Security Modules), targeting high isolation between VMs but without mentioning TPM access for guest OSes. Both proposals lack a solution to standardize the access to TPM functions such as the TEE.

VI. Conclusion and Outlook

This paper proposes T-KVM, a new security architecture for ARM v7 and v8 virtualized systems, providing architecture details and a preliminary performance analysis. T-KVM's architecture offers strong isolation for guest applications by means of the KVM hypervisor, ARM TrustZone, SELinux and a virtualization enabled implementation of the GlobalPlatform TEE API. The last component design is the main contribution of this paper, as it enables the support of Trusted Computing features in ARM based virtual machines through the use of a QEMU device and a shared memory.

The benefits of the proposed solution are its flexibility and compatibility with the existing Cloud and smart devices architectures. In fact, since T-KVM adapts and combines existing opensource components which are already part of virtualized systems and OpenStack Cloud Computing infrastructure (i.e., KVM, libvirt, qemu, etc.), the support of T-KVM in these environments is straightforward to implement. Moreover, the possibility to install a real-time operating system in the TrustZone Secure World adds the support for real-time applications, which is interesting for automotive, avionics and networking applications. Lastly, monitoring and (remote) attestation are eased by the combination of a standard hypervisor with TrustZone.

On the other hand, the lack of the ARM VE support in the Secure World does not allow the hardware assisted virtualization of the TEE. Nonetheless, it is possible to functionally implement multiple TEE (or vTPMs) using paravirtualization or virtualization at the application layer.

As for the measurements and the analysis of the overhead introduced by the proposed solution, we can claim that the performance cost of T-KVM is acceptable. In fact, even if the secure request overhead is significantly higher than a trap in the KVM hypervisor, the execution frequency of the former is expected to be lower than the latter. In fact a service request is issued by a T-KVM guest only to control the communication between the guest and the Secure World, as all the data exchanges will be performed through shared memory.

Finally, the future work includes the implementation of a complete T-KVM ARMv8 prototype. Of interest is also the support and integration of T-KVM in OpenStack, which would enable a complete new set of Cloud features based on Trusted Computing such as location aware scheduling of new instances, trusted multitenancy, etc.

Acknowledgment

This research work has been supported by the FP7 TRESCCA project under the grant number 318036.

 ISBN: 978-1-61208-388-9

REFERENCES

[1] M. Wei, B. Heinz, and F. Stumpf, "A cache timing attack on aes in virtualization environments," in Financial Cryptography and Data Security, ser. Lecture Notes in Computer Science, A. Keromytis, Ed. Springer Berlin Heidelberg, 2012, vol. 7397, pp. 314–328. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-32946-3_23

[2] B. Saltaformaggio, D. Xu, and X. Zhang, "Busmonitor: A hypervisor-based solution for memory bus covert channels," 2013.

[3] G. Pék, A. Lanzi, A. Srivastava, D. Balzarotti, A. Francillon, and C. Neumann, "On the feasibility of software attacks on commodity virtual machine monitors via direct device assignment," in Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, ser. ASIA CCS '14. New York, NY, USA: ACM, 2014, pp. 305–316. [Online]. Available: http://doi.acm.org/10.1145/2590296.2590299

[4] N. Elhage, "Virtunoid: A kvm guest-¿ host privilege escalation exploit," 2011.

[5] S. E. Madnick and J. J. Donovan, "Application and analysis of the virtual machine approach to information system security and isolation," in Proceedings of the Workshop on Virtual Computer Systems. New York, NY, USA: ACM, 1973, pp. 210–224. [Online]. Available: http://doi.acm.org/10.1145/800122.803961

[6] M. Paolino, "Isolating arm platforms: towards secure virtualized embedded systems," May 2014, last visited March 2015. [Online]. Available: http://www.cspforum.eu/uploads/Csp2014Presentations/Track_8/Strong%20Isolation%20in%20ARM%20Platforms.pdf

[7] G. Chen, "Kvm open source virtualization for the enterprise and openstack clouds," IDC, 2014.

[8] C. Dall and J. Nieh, "Kvm/arm: Experiences building the linux arm hypervisor," 2013.

[9] T. Alves and D. Felton, "Trustzone: Integrated hardware and software security," ARM white paper, vol. 3, no. 4, 2004, pp. 18–24.

[10] GlobalPlatform, "Tee client api specification, public release v1.0," 2010.

[11] ——, "Tee internal api specification, public release v1.0," 2011.

[12] X. Xu, C. Xiao, C. Gao, and G. Tian, "A study on confidentiality and integrity protection of selinux," in Networking and Information Technology (ICNIT), 2010 International Conference on, June 2010, pp. 269–273.

[13] A. Jasti, P. Shah, R. Nagaraj, and R. Pendse, "Security in multi-tenancy cloud," in Security Technology (ICCST), 2010 IEEE International Carnahan Conference on, Oct 2010, pp. 35–41.

[14] A. Vahidi and C. Jämthagen, "Secure rpc in embedded systems: Evaluation of some globalplatform implementation alternatives," in Proceedings of the Workshop on Embedded Systems Security, ser. WESS '13. New York, NY, USA: ACM, 2013, pp. 4:1–4:7. [Online]. Available: http://doi.acm.org/10.1145/2527317.2527321

[15] J.-E. Ekberg, K. Kostiainen, and N. Asokan, "The untapped potential of trusted execution environments on mobile devices," Security Privacy, IEEE, vol. 12, no. 4, July 2014, pp. 29–37.

[16] C. Gehrmann, H. Douglas, and D. Nilsson, "Are there good reasons for protecting mobile phones with hypervisors?" in Consumer Communications and Networking Conference (CCNC), 2011 IEEE, Jan 2011, pp. 906–911.

[17] F. Liu, L. Ren, and H. Bai, "Secure-turtles: Building a secure execution environment for guest vms on turtles system," Journal of Computers, vol. 9, no. 3, 2014, pp. 741–749.

[18] GlobalPlatform, "Tee system architecture v1.0," 2011.

[19] M. Paolino, M. M. Hamayun, and D. Raho, "A performance analysis of arm virtual machines secured using selinux," in Cyber Security and Privacy, ser. Communications in Computer and Information Science, F. Cleary and M. Felici, Eds. Springer International Publishing, 2014, vol. 470, pp. 28–36. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-12574-9_3

[20] K. W. Miller, J. Voas, and G. F. Hurlburt, "Byod: Security and privacy considerations," IT Professional, vol. 14, no. 5, 2012, pp. 53–55.

[21] F. Bellard, "Qemu, a fast and portable dynamic translator," in Proceedings of the Annual Conference on USENIX Annual Technical Conference, ser. ATEC '05. Berkeley, CA, USA: USENIX Association, 2005, pp. 41–46.

[22] C. Macdonell, X. Ke, A. W. Gordon, and P. Lu, "Low-latency, high-bandwidth use cases for nahanni/ivshmem," 2011.

[23] M. Paolino, "A shared memory zero-copy mechanism for arm vms:vosyshmem," last visited March 2015. [Online]. Available: http://www.virtualopensystems.com/en/products/vosyshmem-zerocopy/

[24] G. Klein, K. Elphinstone, G. Heiser, J. Andronick, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood, "sel4: Formal verification of an os kernel," in Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles, ser. SOSP '09. New York, NY, USA: ACM, 2009, pp. 207–220. [Online]. Available: http://doi.acm.org/10.1145/1629575.1629596

[25] "Optee," last visited March 2015. [Online]. Available: https://github.com/OP-TEE

[26] "Osv," last visited March 2015. [Online]. Available: http://www.osv.io

[27] J. Brunel, R. Pacalet, S. Ouaarab, and G. Duc, "Secbus, a software/hardware architecture for securing external memories," in Mobile Cloud Computing, Services, and Engineering (MobileCloud), 2014 2nd IEEE International Conference on, April 2014, pp. 277–282.

[28] M. D. Grammatikakis, "System-level modeling of a noc firewall on spidergon stnoc," May 2014, last visited March 2015. [Online]. Available: http://www.cspforum.eu/uploads/Csp2014Presentations/Track_8/System-Level%20Modeling%20of%20a%20NoC%20Firewall%20on%20Spidergon%20STNoC.pdf

[29] "Arm trusted firmware," last visited March 2015. [Online]. Available: https://github.com/ARM-software/arm-trusted-firmware

[30] E. Keller, J. Szefer, J. Rexford, and R. B. Lee, "Nohype: Virtualized cloud infrastructure without the virtualization," SIGARCH Comput. Archit. News, vol. 38, no. 3, Jun. 2010, pp. 350–361. [Online]. Available: http://doi.acm.org/10.1145/1816038.1816010

[31] J. Szefer, E. Keller, R. B. Lee, and J. Rexford, "Eliminating the hypervisor attack surface for a more secure cloud," in Proceedings of the 18th ACM Conference on Computer and Communications Security, ser. CCS '11. New York, NY, USA: ACM, 2011, pp. 401–412. [Online]. Available: http://doi.acm.org/10.1145/2046707.2046754

[32] A. Seshadri, M. Luk, N. Qu, and A. Perrig, "Secvisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity oses," SIGOPS Oper. Syst. Rev., vol. 41, no. 6, Oct. 2007, pp. 335–350. [Online]. Available: http://doi.acm.org/10.1145/1323293.1294294

[33] A. M. Azab, P. Ning, Z. Wang, X. Jiang, X. Zhang, and N. C. Skalsky, "Hypersentry: Enabling stealthy in-context measurement of hypervisor integrity," in Proceedings of the 17th ACM Conference on Computer and Communications Security, ser. CCS '10. New York, NY, USA: ACM, 2010, pp. 38–49. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866313

[34] X. Ge, H. Vijayakumar, and T. Jaeger, "Sprobes: Enforcing kernel code integrity on the trustzone architecture," CoRR, vol. abs/1410.7747, 2014. [Online]. Available: http://arxiv.org/abs/1410.7747

[35] R. Perez and et al., "vtpm: virtualizing the trusted platform module," in Proc. 15th Conf. on USENIX Security Symposium, 2006, pp. 305–320.

[36] R. e. a. Sailer, "shype: Secure hypervisor approach to trusted virtualized systems," 2005.

[37] H. Nahari, "Trusted secure embedded linux," in Proceedings of 2007 Linux Symposium. Citeseer, 2007, pp. 79–85.

[38] T. K. Lengyel, T. Kittel, J. Pfoh, and C. Eckert, "Multi-tiered security architecture for arm via the virtualization and security extensions," 2014, pp. 308–312. [Online]. Available: http://dx.doi.org/10.1109/DEXA.2014.68

 ISBN: 978-1-61208-388-9

Abstraction Layer Based Distributed Architecture for Virtualized Data Centers

Ali Kashif Bashir, Yuichi Ohsita, and Masayuki Murata
Graduate School of Information Science and Technology
Osaka University
Osaka, Japan.
e-mail: {ali-b, y-ohsita, murata}@ist.osaka-u.ac.jp

Abstract— **Network virtualization was envisioned to enhance the capabilities of data centers. However, existing virtual data center network architectures are unable to exploit the features of network virtualization. In this paper, we propose a distributed virtual architecture that groups virtual machines into clusters of different service types. This architecture introduces a concept named *abstraction layer* consisting of virtual switches that are logically grouped together to perform the role of cluster heads. The abstraction layer provides a better control and management of clusters. This architecture enables several features of network virtualization such as scalability, flexibility, high bandwidth, etc. However, in this work, we evaluated the failure of servers and virtual machines to prove the efficiency and scalability of the architecture.**

Keyword-virtualization; infrastructure for clouds; data center network architecture; future internet; scalable data center architecture.

I. INTRODUCTION

Data Center Networks (DCNs) are experiencing a rapid growth in both scale and complexity as they can host large-scale applications such as cloud-hosting. Such growth imposes huge challenges to upgrade the current infrastructure of data centers. However, the current infrastructure is owned by a large number of Internet Service Providers (ISPs) and it is difficult to adopt new architectures without the agreement of all stakeholders.

Virtualization is a technique where the functionalities of server are copied to Virtual Machines (VMs). With server virtualization, we can create multiple logical VMs on top of a single server that can support various applications e.g. VMware [1] and Xen [2]. These VMs can take away the computation from servers. However, a Virtual Network (VN) is a virtual topology that connects devices of the VN or physical network [3]. One of the properties of VN is that links can be added and deleted easily in it.

Network virtualization [4]-[6] can be defined as a technique where multiple VNs are created on top of a physical DCN. It was envisioned to provide several features to the data centers to support several cloud applications. Some of these features are: scalability to network expansion, adaptability to demands of users, and improve network performance in terms of bandwidth and energy, etc. However, existing virtual architectures of DCNs [6]-[10] provide only one or two features at a time and utilize network resources poorly. Therefore, they are unable to exploit most of the features of network virtualization.

Literature work in virtualizing data centers can be divided into centralized and decentralized approaches. The main centralized architectures are SecondNet [7], which provides bandwidth guarantees and CloudNaas [8], which provides support for deploying and managing applications Centralized architectures suffer when network expands. In decentralized approaches, PolyVine [9] and adaptive VN [10] are two worth discussing approaches. Polyvine embeds end to end VNs in decentralized manners. Instead of technical, it resolves the legal issues among infrastructure providers. In adaptive VNs [10], every server is supposed to have an agent. Each server agent communicates with another to make local decisions. This approach is expensive and needs additional hardware. In general, decentralized architectures have obvious advantages over centralized ones. They have no single point of failure, can run multiple applications concurrently, and are scalable and flexible to network changes.

To exploit the features of network virtualization, in this paper, we propose a distributed virtual architecture named A*bstraction Layer Based Virtual Clusters* (AL-VC) for data centers where VMs are grouped into clusters according to their service types. *Abstraction Layer* (AL), used first time in network virtualization architectures, is a key concept of this paper. An AL is created by logically combining a subset of VN switches with an identifier. One AL is assigned to each group of VMs and they jointly form a cluster where AL will perform the jobs of a cluster head.

Introducing AL helps in managing clusters and brings several features to the virtual architecture, such as making AL-VC scalable, adaptable, and flexible. We will discuss these features in the next section. Though AL-VC offers several features, however, in this work, we evaluated its scalability and efficiency in replacing failed VMs or servers.

The rest of the paper is organized as follows: in Section 2, we present the overview of the proposed architecture and discussed its topology, and addressing during routing. Section 3 includes the AL construction algorithm and discusses the features it offers. In Section 4, we present the evaluation of this work and Section 5 concludes the paper.

II. SYSTEM OVERVIEW

Virtual data centers are those where some or all of the hardware of the data center is virtualized. A virtual data center is a collection of virtual resources connected via virtual links. This section discusses the overview of AL-VC. It is important to mention that this work does not provide any VM mapping algorithm. There are several VM mapping

 ISBN: 978-1-61208-388-9

algorithm proposed such as [12] that can be used for VM mapping. Therefore, in this work, we assume that servers are already hosting VMs.

A. Architectual Overview

Virtual Clusters (VCs) are more desirable than physical data center because the resource allocation in VC can be rapidly adjusted to meet changing needs of the users. DCN have high correlations. In data centers, every server provides a set of services and their data usually have a high correlation [12]. VMs hosted by these servers also provide similar servers; therefore, they need to interact with each other frequently to provide services to the users. To take advantage of this, in our approach we group VMs into clusters according to their service types, as shown in Figure 1, where VMs offering Social Networking Services (SNS) form one cluster, VMs offering web services form another one, and so on. Forming clusters according to service types will save search and allocation time of queries [13].

B. Topology

Ideally, VN topology should be constructed in a way that it achieves minimum energy consumption, larger bandwidth without much delay. Minimum energy consumption can be achieved by minimizing the active number of ports and constructing energy efficient routes. Larger bandwidth can be achieved by adding virtual links in the VN and by managing traffic efficiently. Delay can be improved by using efficient routes and by processing data faster at switches. Our architecture is capable to meet these challenges.

The topology of AL-VC is presented in Figure 2, where all the servers in the server racks are connected to one Top-of-the-Rack (ToR) switch. Each server is hosting multiple VMs. To construct VN, we use virtual switches name as Optical Packet Switches (OPSs) as they provide large bandwidth and small energy consumption [14]. They are capable to store, buffer, and can inter- convert electronic and optical packets. Note that TOR switches produce electronic packets and, in order to route those packets over VN, they first need to be converted into optical packets. OPSs send optical packet and they need to be converted before forwarding to TOR switches. An OPS has the tendency to do

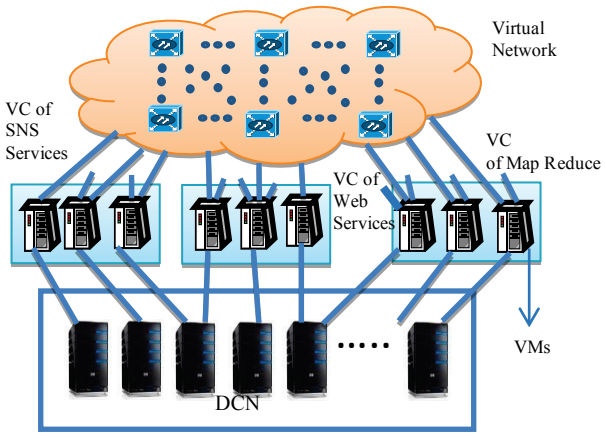


Figure 1. Overview of AL-VC

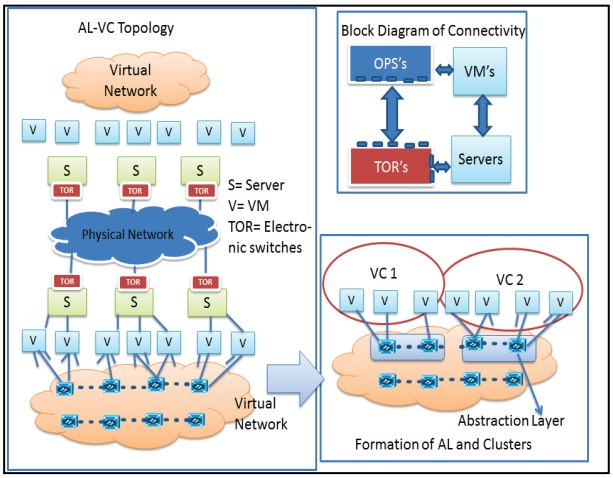


Figure 2. AL-VC Topology

this inter-conversion. In AL-VC, we restricted the communication among VMs only via OPSs. Every VM is connected to multiple OPSs. OPS that joins a particular AL can have four possible types of connections, namely: 1- with TOR switches, 2- with VMs of local cluster, 3- with OPSs of local AL, and 4- with OPSs of VN that are not part of its local AL. In Figure 2, we show the block diagram of this connectivity and as well the logical construction of AL-VC.

C. Addressing

AL-VC is monitored by a central entity called *network manager*. It monitors and controls all resources such as servers, VMs, links, etc. Network manager is responsible for VC formation and deletion. It decides the number of clusters according to service types, sizes of the clusters, and how they are mapped to the servers. It also assigns each VC with a unique VC_{ID} and IP address. However, controlling and managing the cluster after creation is the job of its AL. For address isolation, every VC has its own IP address space. VMs within a cluster communicate with each other via AL.

III. ABSTRACTION LAYER

In this section, we first present our AL construction algorithm and then we discuss the advantages that an AL offers to distributed architectures.

A. Construction of an AL

The basic idea behind the construction of an AL is logically allocating a subset of VN switches to a particular group of VMs. Each switch in an AL knows the topology of its cluster such as VMs locations and their connections.

To construct an AL, VMs of every cluster selects the minimum subset of OPSs that connect all the VMs. This approach selects the switches with highest connections and then switches with second highest connections and so on until all the VMs are connected. The subset of switches that covers all the VMs of a cluster will be declared as its AL. They can be distinguished from other switches of VN with the respective cluster ID. Information of these switches such as switch ID and IP addresses is forwarded to all the VMs. All devices will update their routing tables to identify other

 ISBN: 978-1-61208-388-9

switches of the AL. This procedure is repeated for every cluster until all the clusters have an AL. The detailed mechanism is as follows:

Step 1: After VMs are grouped into clusters, they connect themselves to the switches of VN. These connections can be established randomly or based on a specific criterion. In this work, we use random approach shown in Figure 3. The selection probability of the switches of AL is based on the distance, in which we have

$$P_i = \frac{R_i}{\sum_j d_j} \quad (1)$$

where
P_i = probability of selecting switches v_s^i
d_j = distance of switches from VM

Step 2: Each VM sends a list of the candidate switches they connect to the network manager. Figure 4 (a) shows the list of switches each VM will send to network manager.

Step 3: Network manager selects the minimum set of switches that cover all VMs. To explain this, let's assume a graph $G = (V, E)$ with links $l_i \geq 0$, where the objective is to find a minimum subset of switches that covers all VMs. For this, we apply the following condition to VMs

$$S_i = \begin{cases} 0 & \text{if VM } v_s^i \text{ is not covered} \\ 1 & \text{if VM } v_s^i \text{ is covered} \end{cases}$$

Objective function: minimize $\Sigma 1 S$ for all v_s

Figure 4 (b) is the final minimum set of switches required to form an AL for a cluster. These switches will be announced as an AL for a cluster such as S_1, S_2, and S_3 in Figure 4(b) and are discussed as OPS in this paper. These OPSs will be assigned with VC_{ID}. In routing the traffic, OPSs in the intra-cluster phase can be addressed with (S_{ID}, and IP address) and in inter-cluster phase as (S_{ID}, VC_{ID}, and IP address). Selecting a switch with maximum connections will reduce the number of switches in an AL and it will also help in aggregating the traffic. On the other hand, it may increase load on particular switches. Thus, there is a trade-off that needs to be considered for efficient architectures, however, this objective is not considered in this work.

Step 4: After selecting an AL, the remaining candidate switches will be discarded and they again start acting as ordinary switches of VN.

This procedure is repeated until every cluster has an AL.

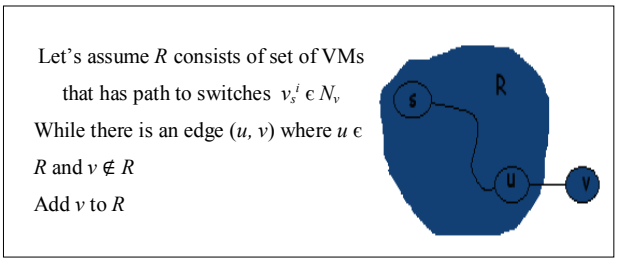


Figure 3. Switch selection criteria

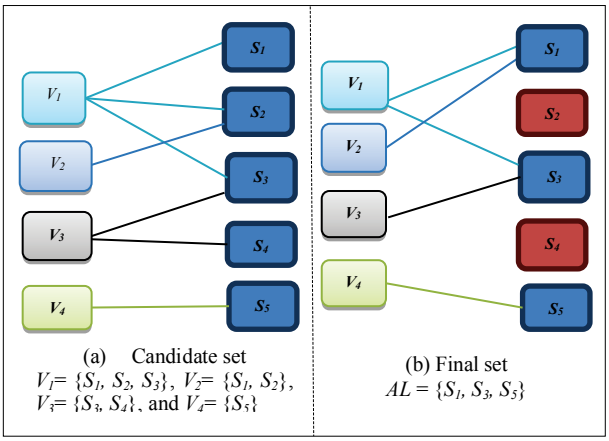


Figure 4. Selection of an AL

B. Features ALs Offer

It is depicted from the literature study [5], [12] that virtualization architecture should be capable to meet the required bandwidth, should be scalable to network changes, should manage traffic efficiently to preserve resources, and should use available resources efficiently to meet the future demands of the users. We claim that AL-VC is a potential architecture to meet these challenges.

In our architecture, an AL provides an abstraction to the clusters. Suppose we group VMs without an AL, then the traffic generated by the VMs is directly routed to the switches of VN. VN switches have to first convert electronic packets coming from TOR switches into optical packets and then route to the destination as shown in Figure 5. Switches near the destination VM have to convert optical packets again into electronics before sending to TOR switches. However, in our architecture, switches of an AL converts TOR packets into optical and then route towards the VN switches. It takes away the computation burden from VN switches and leaves them only for routing data. This allocates more bandwidth for the data. Moreover, due to an AL, we can bisect the traffic into intra-cluster and inter-cluster. When data arrives at an AL, it checks the destination device ID. If the destination machine belongs to its own cluster, AL sends data directly. If destination machine belongs to another cluster, AL will route the data towards the VN. This bisection of traffic provides shorter routes to intra-cluster traffic and let inter-cluster traffic use all the bandwidth of VN switches, which results in lower latency, and higher bandwidth as shown in Figure 5. Below we discuss more features our architecture offers:

Local management and control: Due to ALs, VMs in a cluster can be easily managed and modified without interrupting the operation of the rest of the network. For that, local decisions can be made without the involvement of an external entity.

Scalability: Introducing AL makes clusters quite flexible to network changes. The number of OPS in an AL can vary depending upon the resources of the network resources or the demands of the users. A cluster that has high

 ISBN: 978-1-61208-388-9

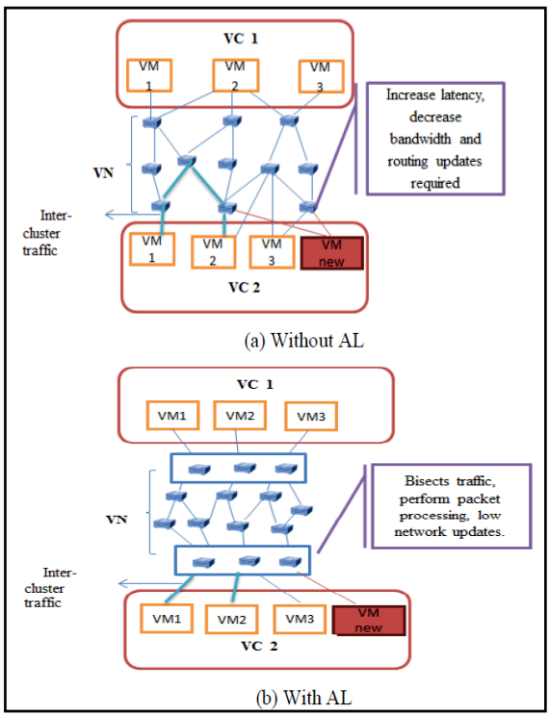


Figure 5. Benefits of an AL

bandwidth demands might need more switches in its AL for faster processing.

Flexibility: AL based VCs are scalable to network expansion and flexible to network failures. In AL-VC, new machines can be added or deleted easily. In case of deletion or failure of machines, AL-VC can replace them with the new ones by local discovery mechanism.

Security: One of the assumptions of this architecture is that one OPS cannot be part of two ALs. OPSs of two different ALs will communicate via intermediate switches of VN. However, within an AL, they communicate directly to process the cluster data jointly. Avoiding direct communication of VMs helps in improving security. VMs can be attacked by intruders when connected to the Internet. Restricting their communication only via OPSs will hide their physical location, hence, will result in a better security.

Implement-ability: Unlike other proposed virtual architectures, AL-VC is implementable on any underlying physical topology of data centers such as on VL 2, B Cube, etc. It basically collects all the virtual resources in a pool and forms VCs according to the requirements of the ISPs, as shown in the Figure 6.

Meeting Application Criteria: VCs should be flexible enough to meet the changing demands of the users. Due to above features, we think, AL-VC is a potential architecture for this purpose. For example, the number of clusters, the number of VMs in a cluster, and the number of switches in an AL can be adjusted to provide the required bandwidth and latency.

All these features make AL-VC a standalone virtualization architecture that tends to exploit most of the features of network virtualization.

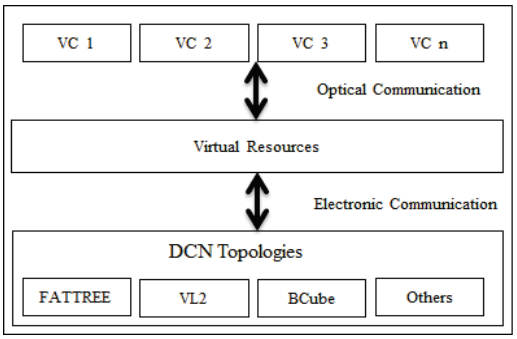


Figure 6. Implementability of AL-VC

IV. PERFORMANCE EVALUATION OF AL-VC

Though AL-VC offers several advantages, it is not possible to evaluate all of them in this work. In this work, we evaluated the scalability and efficiency in recovering from network failures such as VM and server failure. We use centralized architecture as the base scheme. For implementation, we select FATTree [15] as the underlying physical topology.

A. Failure of VMs

When a server detects the failure of a VM or when a VM is not replying to the control messages of its AL, VM is considered as failed. AL will inform all the VMs of its cluster about this failure. After this detection, AL will request the server that was hosting this failed VM to launch a new VM.

If sever does not have enough resources to host a new VM, it will send attributes of the failed VM to the AL. AL will request other servers that have the resources to host a new VM. Servers will send the attributes of candidate VM to AL. AL will select the VM of the server that has the closest attributes to the failed VM. Finally, the failed VM will be replaced with a new VM. The attributes of the requested VM can be represented as:

$$att_{Nv} = ((att_1, n_v^{\,1}), (att_2, n_v^{\,2}), \ldots, (att_n, n_v^{\,n}) \qquad (2)$$

Non-Functional (NF) attributes of the two VMs can be calculated by the following dissimilarity metrics:

$$dism(i,j) = \frac{\sum_{r=1}^{l} \delta_{ij}^{r} dis_{ij}^{r}}{\sum_{r=1}^{l} \delta_{ij}^{r}} \qquad (3)$$

where:

l is the number of NF attributes

dis_{ij}^{r} denotes the dissimilarity of VM i and j related to att_l

δ_{ij}^{r} expresses the coefficients of the NF attributes of VMs i and j.

In Figures 7, 8, and 9, we evaluated the performance of AL-VC in detecting and replacing the failed VM/VMs. Centralized approach uses central fault detection and recovery. First, each server detects the failure and informs the central entity. For that, the central entity exchanges

TABLE I. SIMULATION ENVIRONMENT

Number of Servers	96
Number of VMs	360
Max VM a server can host	10
Number of switches in AL	10 % of VM in the cluster
Number of clusters	2, 4, 6, 8, and 10
DCN topology	FATREE
Parameters	Average time and Communication Cost

messages with all the participating servers to discover a new host. However, in our approach this procedure happens at local AL where AL takes the decision involving local machines. Therefore, AL requires less number of messages and less time to find new a VM to replace the failed VM, as can be seen in Figures 7 and 8.

The average time is the time required to detect a failed VM and replace it with a new VM. The communication cost is a measure of the number of messages required to replace these VMs. From Figures 7 and 8, we can see that an increased number of clusters decreases the average time and communication cost. This is because the number of participating entities in finding a new VM decreases. Increasing number of clusters helps in improving

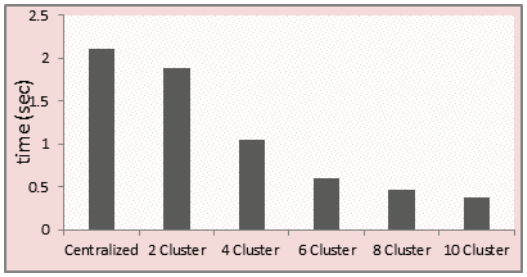


Figure 7. Average time required to replace failed VM.

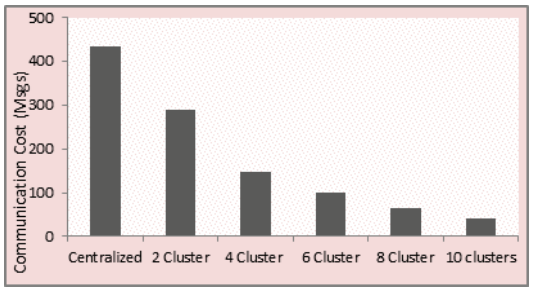


Figure 8. Communication cost required in replacing failed VM

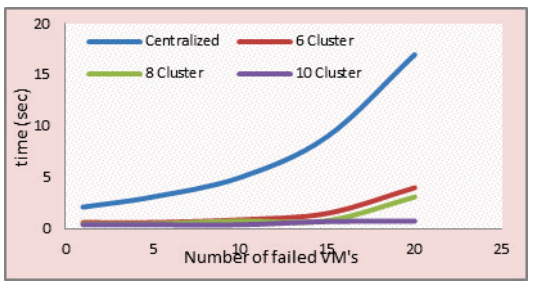


Figure 9. Average time required to replace failed VMs

the performance of our approach. However, too many clusters in the network may result in overhead.

In Figure 9, we measure the time to replace multiple failed VMs. We can clearly see that when the number of failed VMs increases, the performance of the centralized approach deteriorates as the central entity has a lot of the workload and failure of multiple VMs can result in queuing delay at central entity. In case of AL-VC, each AL can run the VM discovery procedure locally to find the new VMs with less overhead.

B. *Failure of Servers*

When a server fails, all the VMs hosted by this server will also go down. When a VM does not respond to keep-alive messages, AL considers it as failed and contacts the hosting this VM. If server also does not respond, AL assumes that the servers has failed or has been removed from the cluster. AL informs to the network manager and asks for the attributes of the failed server and its VMs. After receiving NF attributes, it runs a local VM discovery algorithm to find new hosts for the VMs as explained before.

Note that failure of servers or VMs belonging to one cluster will not affect the operation of other clusters. In this evaluation, we assume that the failing server has three VMs that need to be relocated or replaced. From Figures 10 and 11, we can clearly see that AL-VC takes less time and less number of messages to replace these VMs. If the resources of the cluster are tight, network manager can search for a new server; if new server is available, it can replace the failed server with the new one by matching their nonfunctional attributes. Attributes of the requested server can be represented as following:

$$att_{Ns} = ((att_1, n_s^{\,1}), (att_2, n_s^{\,2}), \ldots\ldots, (att_n, n_s^{\,n}) \quad (4)$$

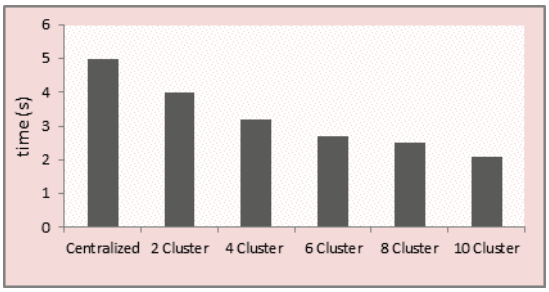


Figure 10. Average time to recover from a server failure

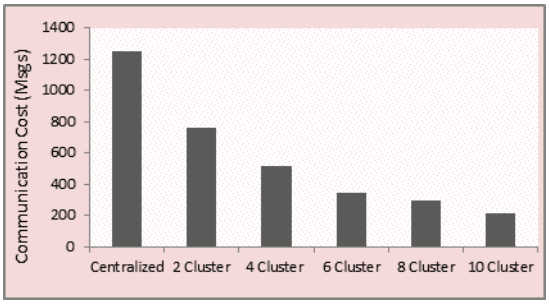


Figure 11. Communication cost required in recovering from a server failure

These results prove that AL-VC is efficient in terms of time and cost. We conducted this evaluation in comparison with centralized approach; however, in extension of this work, we plan to evaluate our architecture with other distributed schemes as well.

V. CONCLUSIONS

Network virtualization is essential for the future Internet. It provides several features to the data centers. Existing virtual data center architectures are not capable to provide all these features. Therefore, in this paper, we proposed AL-VC that groups VMs into cluster based on service types. AL is the main feature of our architecture consisting of virtual switches of the VN. The introduction of AL helps in meeting most of the challenges that network virtualization envisioned such as scalability, flexibility, better control and management, and so on. In this work, we evaluated only its efficiency and scalability in the presence of failures. However, in the future, we plan to evaluate other parameters like bandwidth, latency, etc, as well.

ACKNOWLEDGEMENT

This work was supported by the National Institute of Information and Communications Technology (NICT), Japan.

REFERENCES

[1] WMware, "Virtualization Overview," White paper, pp.1-11, 2006.

[2] T. Abels, P. Dhawan, and B. Chandrasekaran, "An Overview of Xen Virtualization," Virtual. Tech., pp. 109-111, Aug. 2005.

[3] D. Stezenbach, M. Hartman, and K. Tutschku, "Parameters and Challenges for Virtual Network Embedding in the Future Internet," Network Operations and Management Symposium (NOMS 2012) IEEE, Apr. 2012, pp. 1272-1278, ISSN: 1542-1201, ISBN: 978-1-4673-0267-8

[4] N.M.M.K Chowdhury, and R. Boutaba, "Network Virtualization: State of the art and Research Challenges," Communications Magazine, IEEE, vol. 47, pp. 20-26, Jul, 2009, doi: 10.1109/MCOM.2009.5183468

[5] K. Tutschku, T. Zinner, A. Nakao and P. Tran-Gia, "Network Virtualization: Implementation Steps towards the Future Internet," EASST [Online]. Available from: http://journal.ub.tu-berlin.de/eceasst/article/view/216/218

[6] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang, and M.F. Zhani, "Data Center Network Virtualization: a Survey," Communications Survey and Tutorioals, IEEE, vol. 15, pp. 909-928, Sep. 2013,doi: 10.1109/SURV.2012.090512.00043

[7] C. Guo, G. Lu, J. H. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y.Zhang, "SecondNet: a Data Center Network Virtualization Architecture with Bandwidth Guarantees," The Sixth International Conference on Emerging Networking Experiments and Technologies (Co-Next 2010) ACM, Dec. 2010, pp. 1-14, doi: 10.1145/1921168.1921188

[8] T. Benson, A.A.A. Shaikh, and S. Sahu, "CloudNaaS: a Cloud Networking Platform for Enterprise Applications," Second Symposium on Cloud Computing (SOCC'2011). ACM, Oct. 2011, doi: 10.1145/2038916.2038924

[9] M. Chowdhury, F.Samuel, and R. Boutaba, "PolyViNE: Policy-based Virtual Network Embedding across multiple Domains," The Second Sigcomm workshop on Virtualized Infrastructure Systems and Architectures (VISA 2010) ACM, Sept.2010, pp. 49-56, doi: 10.1145/1851399.1851408

[10] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive Virtual Network Provisioning," The Second Sigcomm workshop on Virtualized Infrastructure Systems and Architectures (VISA 2010) ACM, Sept.2010, pp. 41-48, doi: 10.1145/1851399.1851407

[11] I. Houidi, W. Louati, and D. Zeghlache, "A Distributed Virtual Network Mapping Algorithm," The Eigth International Conference on Communications (ICC 2008) IEEE, May. 2008, pp. 5634–5640, doi: 10.1109/ICC.2008.1056

[12] Y. Zhang, A.J. Su, and G. Jiang, "Evaluating the Impact of Data Center Network Architectures on Application Performance in Virtualized Environments," The Eigteenth International Workshop on Qualiting of Service (IWQoS 2010)IEEE, Jun. 2010, pp. 1-5, doi: 10.1109/IWQoS.2010.5542728

[13] A. Fischer, J.F. Botero, M.T. Beck, H.D. Meer, and X. Hesselbach, "Virtual Network Embedding: a Survey," Communication Surveys and Tutorials, IEEE, vol. 15, pp. 1888-1906, Nov. 2013, doi: 10.1109/SURV.2013.013013.00155

[14] Y. Ohsita, and M. Murata, "Data Center Network Topologies using Optical Packet Switches," The Thirty Second Distributed Computing System Workshop (ICDCSW 2012) IEEE, Jun. 2012, pp. 57-64, doi: 10.1109/ICDCSW.2012.53

[15] M. A. Fares, A. Loukissas, and A. Vahdat, "A Scalable, Commodity Data Center Network Architecture," Conference on Data Communication (SIGCOMM 2008) ACM, Oct. 2008, pp. 63-74, doi: 10.1145/1402958.1402967

[16] N. Farrington and A. Andreyev, "Facebook's Data Center Network Architecture," Facebook, Inc [Online]. Last access: Mar, 2015. Available from: http://nathanfarrington.com/papers/facebook-oic13.pdf .

[17] K. Chen, A. Singh, and K. Ramachandran, "OSA: an Optical Switching Architecture for Data Center Networks with Unprecedented Flexibility," Networking Transactions, IEEE/ACM, vol. 22, pp. 498-511, Apr.2014, doi: 10.1109/TNET.2013.2253120

 ISBN: 978-1-61208-388-9

Cloud Storage Prediction with Neural Networks

Stefan Frey, Simon Disch, Christoph Reich, Martin Knahl
Furtwangen University
Cloud Research Lab
Furtwangen, Germany
{Stefan.Frey, Ch.Reich, Martin.Knahl}@hs-furtwangen.de

Nathan Clarke
Plymouth University
Centre for Security, Communication and Network Research
Plymouth, United Kingdom
cscan@plymouth.ac.uk

***Abstract*—In this work, we present an Artificial Neural Network approach to predict the usage, size and type of a cloud storage to enable better compliance with Service Level Agreements (SLAs). One of the biggest advantage of cloud infrastructures is scalability on demand. Cloud services are monitored and based on utilization and performance need, they get scaled up or down, by provision or deprovision of resources. The goal of the presented approach is to predict and thereof select the right amount of storage with a minimum of preallocated resources, as well as the corresponding storage type based on the predicted performance needs in order to reduce SLA violations. Evaluation of the results obtained by simulation confirm that, by using this approach, SLA violations decreased compared to a threshold value control system.**

Keywords–Cloud Computing, Storage, Prediction, Neural Networks, SLA, QoS

I. INTRODUCTION

After an initial hype, cloud computing has established itself as an adequate means of providing resources on demand on a self-service basis and gives customers access to a large pool of computational power and storage. With cloud computing, customers do not have to manage and maintain their own Information Technology (IT) assets and are not bound to their locally limited resources. In order for both customers and providers to be confident that their cloud services are usable at an adequate level, Quality of Service (QoS) guarantees are needed [1]. For this, service requirements stated in Service Level Agreements (SLAs) need to be monitored and the corresponding resources need to be managed. Currently, cloud providers typically support very simple metrics such as availability, or global best effort guarantees.

In cloud systems, resources are being provided dynamically, which means the quality of a service can be directly dependent on the provisioning mechanism [2]. In order to improve the QoS for cloud computing services QoS monitoring, provisioning strategies, as well as detection and prediction of possible SLA violations must be investigated. In this paper, an approach is proposed to regulate cloud storage through the use of Artificial Neural Networks (ANN). Artificial Neural Networks are computational structures modeled after the biological processes of the brain. According to the Defense Advanced Research Projects Agency (DARPA) Neural Networks are systems composed of many simple processing elements operating in parallel whose function is determined by the network structure, connection strengths, and the processing performed at the elements or nodes [3]. Neural networks have been successfully used for decision support systems and show high potential for the use in forecasting and prediction systems [4]. If one could predict the usage of a service, looking ahead further than the provisioning delay time, one could guarantee the QoS for that specific service. The approach presented in this paper aims to improve SLA compliance through the prediction of cloud storage usage. This addresses particularly the storage allocation and dynamic storage capacity guarantees specified in SLAs.

The remainder of the paper is organized as follows. In Section II, the related research efforts are discussed. Section III presents the cloud QoS model and external factors. In Section IV, the specific approach neural networks for controlling the storage of cloud services is introduced. The proof of concept is reported in Section V. Finally, a conclusion is drawn and future work is suggested in Section VI.

II. RELATED WORK

Neural Networks are widely used in forecasting problems. One of the earliest successful application of ANNs in forecasting is reported by Lapedes and Farber [5]. They used a feedfoward neural network with deterministic chaotic time series generated by the Glass-Mackey equation, to predict such dynamic nonlinear systems.

Atrificial Neural Networks are proven universal approximators [6][7] and are able to forecast both linear [8] and nonlinear time series [9]. Adya and Collopy investigated in the effectiveness of Neural Networks (NN) for forecasting and prediction [4]. They came to the conclusion that NN are well suited for the use of prediction, but need to be validated against a simple and well-accepted alternative method to show the direct value of this approach. Since forecasting problems are common to many different disciplines and diverse fields of research, it is very hard to be aware of all the work done in this area. Some examples are forecasting applications such as: temperature and weather [10][11][12], tourism [13], electricity load [14][15], financial and economics [16][17][18][19] and medical [20][21] to name a few. Zhang, Patuwo, and Hu [9] show multiple other fields where prediction by ANN was successfully implemented.

III. CLOUD STORAGE QOS

SLAs specify the expected performance characteristics between service providers and customers. The most important component of an SLA is the exact description of the service quality (service levels). These descriptions are called Service Level Objectives (SLOs), which contain Key Performance Indicators (KPIs) consisting of metrics and the specific value

 ISBN: 978-1-61208-388-9

to be guaranteed. These metrics are constantly monitored and the SLOs are guaranteed over a relatively long time interval. If the guaranteed service levels are not met, the SLA is violated and penalty costs may have to be paid to the customer by the provider. For storage, typical KPIs stated in an SLA can be the read- and write-speed, storage capacity, random input/outputs per second (IOPS) and bandwidth. Since cloud computing resources can be allocated dynamically at runtime additional, dynamic service level objectives arise. For example, this could comprise a constant growth of the storage capacity or the compliance with a certain maximum deployment time or guarantee a constant minimum of available free memory.

Cloud storage resources are usually multi tenant, which means for the provider that it can economically be very important to distribute the storage as efficiently as possible. This means allocating as close to the minimum guaranteed amount of storage as possible. In practice, this can lead to problems because the memory usage of clients can vary greatly and therefore SLA violations can happen easily. For this, a method shall be found that allows to determine the needed amount of memory close to the optimum and allocate it ahead of time. With such an efficient provisioning method it would be possible for providers to maximize the usability of their infrastructure while at the same time guarantee customers a high quality service.

IV. Artificial Neural Network

The aim of this work was to create a prototype application which enables efficient provisioning of cloud storage resources with the use of Artificial Neural Networks to achieve better compliance with SLAs. The most common type of ANNs used for forecasting is the feedforward multilayer perceptron (ffMLP), as seen in Figure 1.

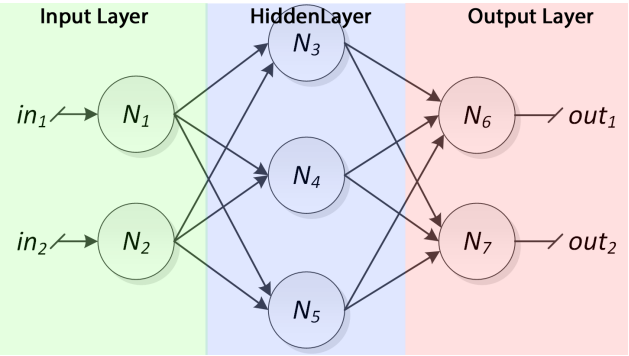


Figure 1. Simple 3-tier Feedforward Multilayer Perceptron.

These are Neural Networks, which consist of one input layer, n-hidden processing layers and one output layer. Feedforward networks are classified by each neuron in one layer having only direct connections to the neurons of the next layer, which means they have no feedback. In feedforward multilayer perceptrons, a neuron is often connected to all neurons of the next layer, which is called completely linked. So, there is no direct or indirect connection path from neuron N_x which leads back to a neuron N_{x-z}. To compute a one-step-ahead forecast, these NNs are using lagged observations inputs of time series or other explanatory variables.

For the creation of the Neural Network model we used the graphical editor and simulator MemBrain [22]. The presented Neural Network consists of 119 neurons, which are aligned into 5 layers, and corresponds to a ffMLP where not all neurons are completely linked. An architectural overview of the presented model is shown in Figure 2 below.

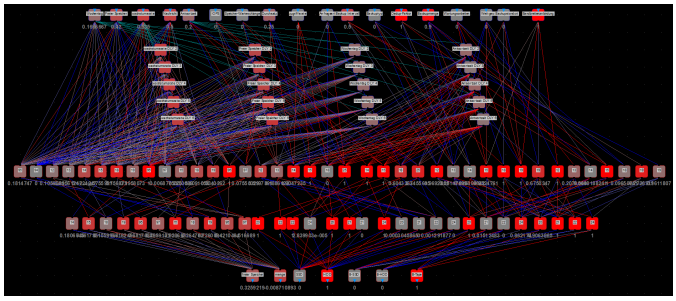


Figure 2. Feedforward Multilayer Perceptron Architecture.

Training of ANNs can be seen as a complex nonlinear optimization problem, and sometimes the network can get trapped into a local minimum. ANNs can theoretically learn by developing new or deleting existing connections, changing the connection weights or threshold values, altering one or more of the three neuron functions (activation, propagation and output) and developing new or deleting existing neurons. In order to improve outputs, the input neurons should get normalized variables. This can simply be done by the equation below.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{1}$$

In order to avoid local minima and bad results, the training should be initialized several times with different starting weights and alignments. For the training of the proposed model, data sets were created in the form of Comma Separated Value (CSV) files. Each file contains storage usage patters with input and output vectors. Here, 60% of the samples were used for training and the remaining 40% were used for the validation of the network. The output behavior was modeled by depending on a input vector, where the desired output values where manually entered into the input vector. Thus, situations in which the responsible output neuron shall increase the amount of allocated memory were mapped.

To teach the network the prediction capability of future memory usage, the input vector was extended. The entire course of the used memory amount was added for the period of t_0 to t_n. The desired output for this input vector at the given time t_i shall be the predicted amount of memory used at time t_{i+x}. To achieve this, the value of the output vector at any point t_i in the time period t_0 to t_n was set to the input vector of the point t_{i+x}, by which x determines the length of the forecast period. Through this shift in values the network can be trained for a prognosis. During each training session the network error was checked with validation data. MemBrain calculates this using the following formula:

$$NetError = \frac{\sum_{i=1}^{n} (Target_i - Output_i)^2}{n} \tag{2}$$

 ISBN: 978-1-61208-388-9

The desired activation of the output neurons is here referred to as $Target$ and the actual calculated activation is the $Output$. The squared deviations are summed and divided by the number of training data sets. To determine whether the Neural Network shows good results of the output behavior, it has been trained and validated with 10 different training data sets. The result for the network error after each learning processes is shown in Table I below.

TABLE I. INFRASTRUCTURE SENSOR PARAMETER.

TrainingNr.	NetError(Training)	NetError(Validation)
1	0,0000573	0,046
2	0,0000586	0,040
3	0,0000713	0,040
4	0,0000702	0,112
5	0,0000611	0,040
6	0,0000783	0,083
7	0,0000703	0,046
8	0,0000627	0,038
9	0,0000645	0,061
10	0,0000630	0,046

Here, it can be seen that the NetError reaches overall good values close to zero and not only for a particular dataset. The average total error for all training runs from Table I is 0.0000657 for trained and 0.0573 for untrained (unknown) input data.

V. EVALUATION

The aim of this work was to investigate, whether or not the use of a Artificial Neural Network for the provisioning of a cloud storage resources has a positive effect on SLAs compliance, and whether this can lead to a better resource utilization compared to a classic threshold value system. For this purpose we created a simulation environment where storage requests (read, write, and delete) form a generator were sent trough a QoS monitor. Inside the QoS control module, the Artificial Neural Network and the threshold value system were used to regulate the amount of allocated storage capacity. Figure 3 shows the architectural overview of the simulation environment.

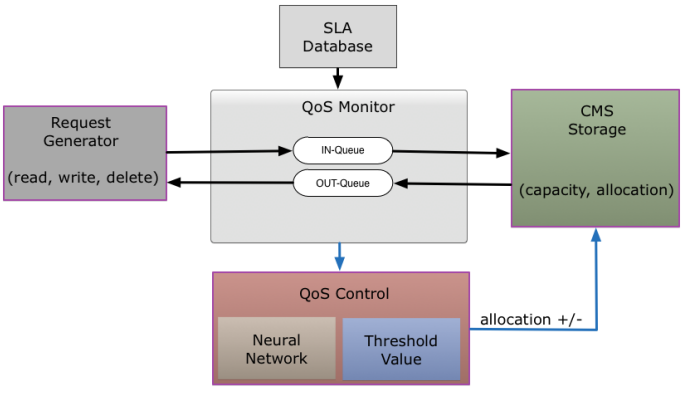


Figure 3. Simulation Architecture.

In the simulation, the impact of regulatory mechanisms on the following key performance indicators was considered:

- Free memory amount: providing an optimal amount of memory by the control logic.
- Response time: compliance with the KPI response time by adjusting the storage medium.
- Backup Media: proposal of a suitable backup medium.

For this, the used Neural Network consisted of 11 different input neurons. Table II lists the used input neurons and describes the used input factors. As output neurons, there is one neuron that gives the expected used memory amount for the next simulation step, a neuron that determines the amount of memory to be added or removed, as well as other neurons that recommend the optimal backup medium.

TABLE II. SIMULATION INPUT NEURONS.

Neuron	Description
Time	Point t_i in $t_0 \ldots t_n$
Weekday	Day of week for point t_i
Free Storage Capacity	Free storage capacity at point t_i
Growth Rate	Change of capacity from t_{i-1} to t_i
Response time	Mean response of last 5 inputs $\frac{\sum_{i=i-5}^{i} t_i}{n}$
Queue Length	Still open request at point t_i
Troughput	Troughput at point t_i
Access Rate	Amount of requests per time slot
Request Type	Distinction between large and small requests
Backup Amount	Size of backup data
Bandwidth	Usable bandwidth at point t_i

In order to compare the results of the Neural Network with a common, in practice widely used method, a threshold value based scaling was implemented. This regulation system is controlled by predefined thresholds for the monitored KPI values. The implementation for the threshold rules for adding and removing allocated storage can be seen below in Figure 4, as simple pseudocode if then rules.

```
// addStorage
IF AllocatedCapacity      UsedCapacity < SLAFreeCapacity +2
THEN
        IF (UsedCapacity < 20)     AllocatedCapacity += 10;
        ELSE IF (UsedCapacity > 80)     AllocatedCapacity += 20;
        ELSE    AllocatedCapacity += 15;

// removeStorage
IF AllocatedCapacity      UsedCapacity > SLAFreeCapacity +15
THEN
        IF (UsedCapacity < 20)     AllocatedCapacity -= 20;
        ELSE IF (UsedCapacity > 80)     AllocatedCapacity -= 10;
        ELSE    AllocatedCapacity -= 15;
```

Figure 4. IF THEN rules for threshold system.

Here, it can be seen that, by falling below a 2% buffer of the storage value defined in the SLA, the allocation will be increased and by exceeding 15% over the amount of storage defined in the SLA, the allocation will be lowered. The amount of which the allocated storage will be changed is dependent on how much the overall storage usage is. In case of an usage of over 80 %) increase will be 20%, with an usage of below 20% the increase will be 10% and in between the increase is 15 % of the overall volume. These settings are reversed for the deallocation of the storage.

For the scenario in this simulation, a dynamic storage SLA, in which a customer gets granted 10GB of free space and up to

100GB of overall usage, was assumed. With such a dynamic limit described in the SLA, it is particularly important for the provider to find a solution that is as close as possible to the guaranteed amount of storage, since this will ensure a high economic efficiency. In practice, however, this usually is not possible. For this reason and because a violation of the SLAs can have monetary consequences, bigger buffer zones are installed. Figure 5 shows the resulting graph of the simulation with the conventional threshold value rules.

Figure 5. Storage allocation results for threshold rules.

The red graph in Figure 5 and Figure 6 shows the course of the memory usage in GB, by the user during the simulation. The usage has been pre-generated for the simulation purpose and shall resemble a system, where a user regularly creates and deletes files with up to 15GB size, as well as generate larger files with up to 50GB. This type of usage may occur while working with different media files, like in the post-processing of movie projects. The green line marks the guaranteed amount of storage available to the user, granted by the SLA. It proceeds synchronous to the red graph, since the user gets guaranteed 10GB more than they currently use. The blue graph shows the pre-allocated amount of storage, which is directly usable by the user.

If we compare these results with those obtained by the Neural Network controlled storage allocation, shown in Figure 6, it becomes clear that the efficiency is marginally improved. With an average of 18.68% of memory over provisioned the threshold value system is almost as effective as the neural network, with a 18.22% overhead. The slight difference arises from the fact that the allocation offered by constantly adopting, fits to the SLA limits with a relatively constant overhead. In contrast, the threshold value system initially provides too much memory, and then only adopts the amount of allocated storage shortly before a violation of the SLA it to occur.

While comparing the two graphs, we see that the threshold system due to the fixed thresholds less often adjusts the amount of memory (blue curve). Since the added / removed amount of memory operates with a fixed predefined value, often too much memory is provided and then immediately gets removed again. This happens likewise when reducing the amount of memory allocated, which often leads to falling below the specified minimum amount in the SLA. However, the Neural Network determines constantly, based on the learned training data, a variable amount of memory that is to be added or removed, which leads to adequate reactions and a slightly better economic result.

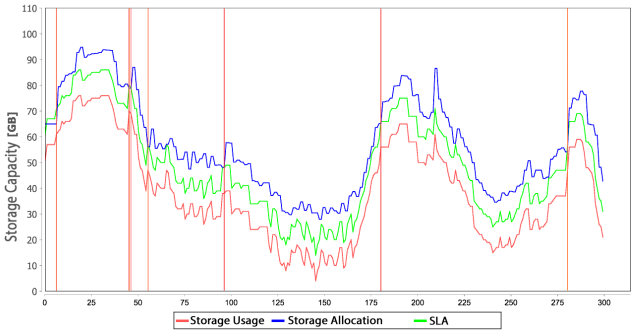


Figure 6. Storage allocation results for NN.

However, when comparing the number of SLA violations, it becomes clear that the Neural Network approach delivers a significantly better solution. This is also evident in the resulted graph seen in Figures 6 and 5, where the SLA violations are indicated by vertical red lines. These exemplary results of the simulation show that the Neural Network produces 7 and the threshold value system 13 SLA violations. These results were also confirmed within the other test runs, where the Neural Network generates an average of 7.45 violations per run and the threshold values system of 15.03 SLA violations per run.

Overall, the Neural Network generated solution for the provisioning of storage is better suited, since the number of SLA violations is significantly lower. Together with the slightly lower overhead makes this a reasonably good solution.

VI. Conclusion and Future Work

The aim of this paper was to investigate whether QoS parameters of a cloud computing storage could be more effectively managed using the predictive capabilities of neural networks. In particular, this study sought to improve the overhead amount of pre-allocated storage and reduce SLA violations. For this, a feedforward multilayer perceptron Artificial Neural Network was presented and its structure and functionality has been delineated. As proof of concept, several tests were performed in order to prove the effectiveness of the approach compared to a threshold value system.

It was found that the over-provisioning of the allocated storage amount could be improved by 0.46% with the ANN prediction. Here improvements with respect to an optimization of the provisioning amount should be carried out. In terms of the SLA compliance, the presented approach significantly won over the threshold value system with nearly halve as many violations. These results shed a positive light on the presented approach, which could lead to an increase in efficiency and economics of cloud storage.

Future work will seek to evaluate this approach within a real cloud environment and with real life differentiating user work loads. Research will also investigate other QoS parameters to understand and improve upon the prediction of cloud storage. Furthermore, the one-step-ahead prediction capability of the used Neural Network should be stretched ahead further into the future in order to improve the forecast and adaptability.

 ISBN: 978-1-61208-388-9

Finally, a deeper evaluation against other prediction methods (e.g Bayesian or Markov Models) is needed, to determine whether or not ANN present the best approach.

ACKNOWLEDGMENT

This research is supported by the German Federal Ministry of Education and Research (BMBF) through the research grant number 03FH046PX2.

REFERENCES

[1] P. Patel, A. Ranabahu, and A. Sheth, "Service level agreement in cloud computing," UKPEW, 2009.

[2] R. Calheiros, R. Ranjan, and R. Buyya, "Virtual machine provisioning based on analytical performance and qos in cloud computing environments," in Parallel Processing (ICPP), 2011 International Conference on, Sept 2011, pp. 295–304.

[3] DARPA Neural Network Study (U.S.), DARPA Neural Network Stdy, Widrow, Morrow, and Gschwendtner, Eds. AFCEA Intl, 1988.

[4] M. Adya and F. Collopy, "How effective are neural networks at forecasting and prediction? a review and evaluation," Journal of Forecasting - Special Issue: Neural Networks and Financial Economics, vol. 17, no. 5-6, September 1998, pp. 481–495.

[5] A. Lapedes and R. Farber, "Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling," Los Alamos National Laboratory, Los Alamos, NM, Tech. Rep. LA-UR-87-2662, 1987.

[6] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Networks, vol. 2, no. 5, 1989, pp. 359–366.

[7] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural Networks, vol. 4, no. 2, 1991, pp. 251–257.

[8] G. P. Zhang, "An investigation of neural networks for linear time-series forecasting," Computers and Operations Research, vol. 28, no. 12, 2001, pp. 1183–1202.

[9] G. Zhang, B. E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of the art," International Journal of Forecasting, vol. 14, no. 1, 1998, pp. 35–62.

[10] G. Langella, A. Basile, A. Bonfante, and F. Terribile, "High-resolution spacetime rainfall analysis using integrated {ANN} inference systems," Journal of Hydrology, vol. 387, no. 34, 2010, pp. 328–342.

[11] J. Taylor and R. Buizza, "Neural network load forecasting with weather ensemble predictions," Power Systems, IEEE Transactions on, vol. 17, no. 3, Aug 2002, pp. 626–632.

[12] P. J. Roebber, M. R. Butt, S. J. Reinke, and T. J. Grafenauer, "Real-time forecasting of snowfall using a neural network," Weather and Forecasting, vol. 22, no. 3, 2014/07/15 2007, pp. 676–684.

[13] D. C. Pattie and J. Snyder, "Using a neural network to forecast visitor behavior," Annals of Tourism Research, vol. 23, no. 1, 1996, pp. 151–164.

[14] D. Park, M. El-Sharkawi, I. Marks, R.J., L. Atlas, and M. Damborg, "Electric load forecasting using an artificial neural network," Power Systems, IEEE Transactions on, vol. 6, no. 2, May 1991, pp. 442–449.

[15] H. Hippert, C. Pedreira, and R. Souza, "Neural networks for short-term load forecasting: a review and evaluation," Power Systems, IEEE Transactions on, vol. 16, no. 1, Feb 2001, pp. 44–55.

[16] Y. Bodyanskiy and S. Popov, "Neural network approach to forecasting of quasiperiodic financial time series," European Journal of Operational Research, vol. 175, no. 3, 2006, pp. 1357–1366.

[17] P. McAdam and P. McNelis, "Forecasting inflation with thick models and neural networks," Economic Modelling, vol. 22, no. 5, 2005, pp. 848–867.

[18] I. Kaastra and M. Boyd, "Designing a neural network for forecasting financial and economic time series," Neurocomputing, vol. 10, no. 3, 1996, pp. 215–236, financial Applications, Part II.

[19] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," Expert Systems with Applications, vol. 38, no. 8, 2011, pp. 10 389–10 397.

[20] A. M. Vukicevic, G. R. Jovicic, M. M. Stojadinovic, R. I. Prelevic, and N. D. Filipovic, "Evolutionary assembled neural networks for making medical decisions with minimal regret: Application for predicting advanced bladder cancer outcome," Expert Systems with Applications, 2014.

[21] C. Arizmendi, D. A. Sierra, A. Vellido, and E. Romero, "Automated classification of brain tumours from short echo time in vivo mrs data using gaussian decomposition and bayesian neural networks," Expert Systems with Applications, vol. 41, no. 11, 2014, pp. 5296–5307.

[22] T. Jetter, "Membrain neural network editor and simulator," [Online] Available: http://www.membrain-nn.de Retrived: 9 August 2014.

 ISBN: 978-1-61208-388-9

A Cloud Application Portal for the Maritime Industry

Bill Karakostas, Takis Katsoulakos
Inlecom Systems Ltd
London, UK
{bill, takis}@inlecom.com

Stelios Christofi
EBOS Technologies Ltd
Nicosia, Cyprus
stelios@ebos.com.cy

***Abstract*— We propose that enterprise portals and cloud management environments should converge, and that such convergence should take place from a user centric perspective. This paper describes the rationale, architecture and implementation of a system that provides user centred oriented integration of cloud applications. The novelty of the approach is that applications are not presented to the user in terms of their Cloud Application Programming Interface (API), but according to the users' task models. This gives the users fine grained control of the applications available to them independently of how they are delivered over the Cloud. We demonstrate the above concepts with a case study from the maritime domain.**

Keywords- Enterprise portal; Cloud portal; Cloud integration; Cloud API; maritime applications.

I. INTRODUCTION

An enterprise portal is an application that gives users a single point of access to other applications and information they need, with the ability to personalize the interface and content as they wish. In this way, a portal can help users make faster, more informed decisions. New generation of enterprise portals are geared toward supporting applications that are hosted in the cloud. In other words, enterprise portals are evolving from information gateways into all-purpose Web platforms that support a wide range of business processes.

The challenge in developing Cloud portals lies in the ability to integrate diverse cloud services into a single user environment that preserves the users' effectiveness to carry out tasks under their domain of expertise, by virtualising and integrating the APIs of diverse cloud services. Current cloud management environments do not support that, assuming the users' familiarity with the API of the cloud application/service.

The system illustrated in this paper integrates virtualised applications in a cloud environment. Applications running on heterogeneous clouds can be accessed from a single portal through a virtualised interface that corresponds to the user domain tasks.

The described system has been developed specifically for the needs of the maritime industry. However, the concepts and approach are equally applicable to other industries that utilize Cloud environments such as retail, finance, and other service operations. The structure of the paper is as follows. The next section surveys the state of the art in enterprise portals and cloud management environments. Section III discusses the research challenges and architecture of the proposed approach. Section IV presents a prototype that applies the proposed approach to the maritime application domain. Finally, section V discusses future plans for further research and extensions to the proposed idea.

II. LITERATURE SURVEY

A. Enterprise Portals

Many enterprise portals today enable website and application development, document management and collaboration. Several portals now integrate with content management systems, or otherwise offer content management functionality and integration with mobile devices, rich media and social media. However, the external applications accessed by such portals, are increasingly been delivered as services on some Cloud, something that has been coined the *cloudification* of legacy applications in Yu et al [1]. Thus, the problem of integrating external applications into an enterprise portal environment, these days, equals to the problem of connecting to and integrating services delivered by heterogeneous clouds.

The importance of Cloud service management through portals has also been acknowledged by the European Union Information and Communications Technology (ICT) Research Programme, which has funded number of projects in the ICT space of APIs, mashups and marketplaces, such as projects Multi-Modal Situation Assessment and Analytics Platform (mOSAIC) and Morfeo 4CaasT.

B. Cloud Management Environments

Although there is some convergence towards the use of communication protocols, i.e. with Resource State Transfer (REST) over http been the preferred method, and, less frequently, Simple Object Access protocol (SOAP)/Web Service Description Language (WSDL), as an alternative method), cloud computing services have diverse APIs through which they can be accessed. A unified interface to provide integrated access to cloud computing services is non-existent, currently. Thus, integration of cloud services into a unified web-based user interface must be carried out by the portal. Virtualisation technologies have a significant role to play towards this goal. Not only the Cloud services

 ISBN: 978-1-61208-388-9

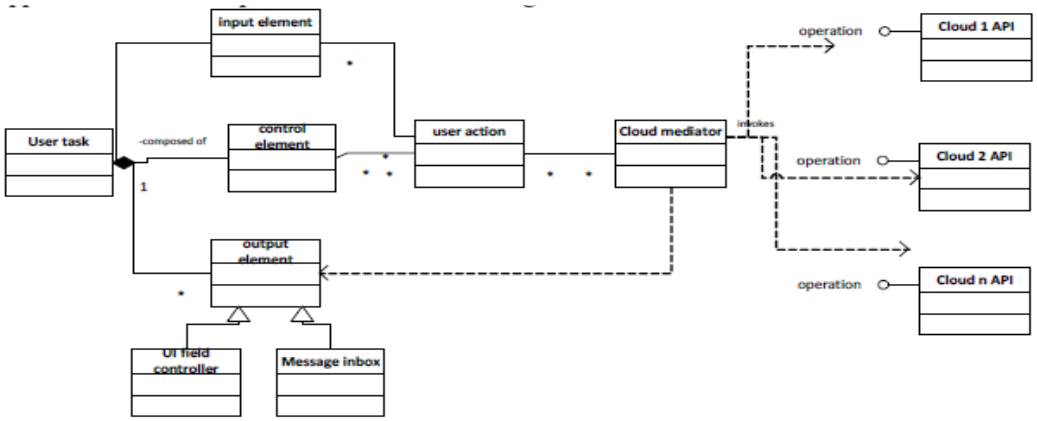


Figure 1: Architecture of mediator-based virtualization of cloud services

virtualise the computing environment, but the portal must provide a user oriented virtualisation environment for the service interfaces. This will virtualise the service interfaces in terms of sets of tasks commonly performed by the user. Some research towards personalized user environments already exists. For example, a paper by Zhang et al [2] presents such a system for Windows based on user-level virtualization technologies. At run-time, the user applications stored on a portable device run in a user-mode virtualization environment where some resource (registry, files/directories, environment variables, etc.) accessing APIs are intercepted and redirected to the portable device as necessary.

Users can access their personalized applications and data on any compatible computer, without the applications actually been installed on that computer. Cloud management environments as proposed in Alrokayan and Buya [3] place emphasis on ease of use, however not necessarily at the user task level, by providing a single point from which the user can control different cloud environment. For example, the Simple Heterogeneous Inter-Cloud Manager (SHINCLOM) project by Powell et al [4] is a prototype web-based single sign-on inter-cloud management portal that gives users the ability to easily configure and launch inter- architecture cloud applications and services. However, most current approaches do not adopt user task oriented strategies for cloud service management. Thus, we propose that enterprise portals and cloud management environments.

III. ARCHTECURE

A Research challenge

The main research challenge in this project is how to integrate heterogeneous cloud environments into a unified portal, in a user centric fashion, as per Gmelch's proposal [5]. The aim is to present applications to the user not according to their native Cloud APIs, but according to the user's task model. This gives the users fine grained control of the applications available to them, independently of how they are delivered over the different Clouds. In contrast, our approach virtualises the cloud application's API in terms of the user's task model. Thus, the user interacts with the Cloud application/service through a virtual API that is converted on the fly to the native API of the application. The approach is explained in the following sections.

B Application Integration Patterns

In this research, we opted for a mediator based integration approach, as per Hohpe and Woolf [6], as this allows the mediation between what the users perceives as the characteristics of their tasks, and what the cloud services offers to them through their APIs. The matching between the two (as well as any mismatches) is handled by the mediator. A task analysis method is used to identify tasks the user performs in the context of an application, the inputs output and control elements to the tasks. The modelling of tasks identifies the information elements that need to be provided by the application. A mapping is performed of the user task's input output and control elements to the Cloud application's/service API operations. As illustrated in Figure 1, the portal's mediator matches user actions and user visible content to the input and output parameters of the external cloud service. At runtime, the mediator will collect user input, convert it to the corresponding API operation and submit to the Cloud service. Depending on the style of interaction (i.e. synchronous vs asynchronous), service response will be mapped to the output user interface elements and immediately displayed to the user, or collected as a message to be delivered to the user's inbox.

C .Dynamic User Interface-Synchronous Communication

In this option, the Cloud application/service has to operate in a synchronous mode. The portal administrator configures the endpoint universal resource locator (URL) of the service. When a user wants to utilize the application, the Portal is responsible for dynamically creating a user interface, using the cloud service interface description (WSDL), and return to the user a data entry form to be used for the interaction with the service. Thus, communication

 ISBN: 978-1-61208-388-9

with the service is synchronous and the response to the user is presented immediately after the submission of the request.

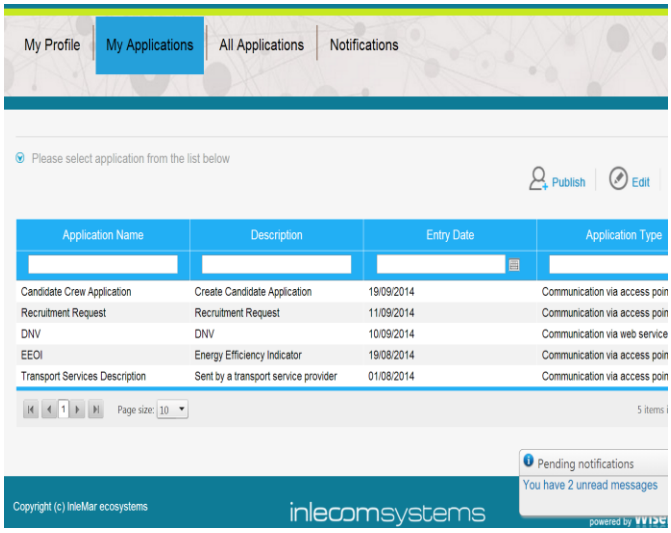


Figure 2: List of user applications

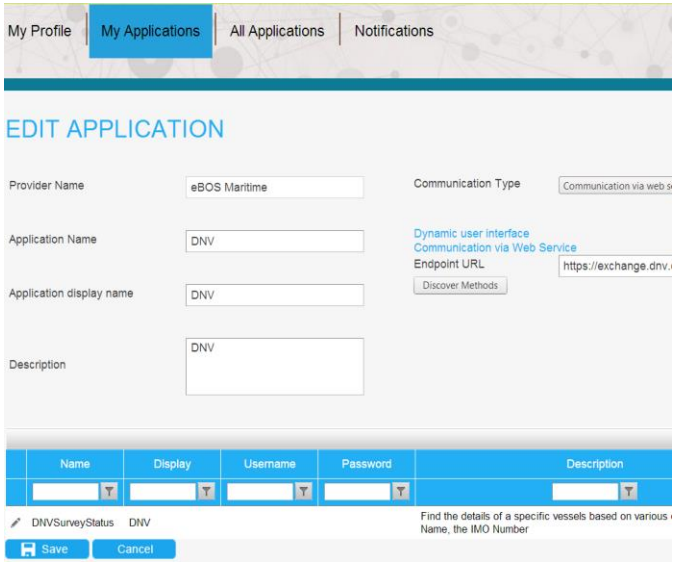


Figure 3: Application configuration by the portal administrator

D.Dynamic User Interfaces-Communications through Access Points

In asynchronous communications, the portal provides a messaging facility to allow users to communicate with the Cloud application. The portal also provides a message queuing facility for applications to exchange messages with the portal. The portal's messaging system handles in-order and exactly-once delivery, required for a consistent and complete presentation of service response messages to the users' inboxes. The portal administrator configures the XML Schema (XSD) of the message that will be exchanged and also how unique identifiers will be generated for its message.

When a user invokes the cloud service, the Portal dynamically creates a user interface using the provided XSD, and returns to the user a data entry form to be used for the interaction with the service. When the service returns a response, a notification is presented to the user's workspace environment.

User applications are configured by the portal administrator who specified the user visible parts of the application as well as the characteristics of the external cloud service (Figure 3). This allows user interfaces to be auto-generated as shown in Figure 4.

IV. PROTOTYPE

As explained already, the emphasis of the described prototype portal is on user centricity. Each user is given a set of applications to work with, organized into categories that are intuitive in the user's domain (i.e. maritime) and role specialization (for example, as recruiter of ship crew). Every time users log in to the portal they are presented with the list of available applications, as shown in the screenshot of Figure 2. With a single sign on to the portal, users can access all applications assigned to them. From their home page, users also have the ability to search for a particular application based on various fields such as the application name, description etc. Moreover, users can customise their view of existing applications. As shown in Figure 2, the users receive notifications from applications they have invoked in an asynchronous manner.

As illustrated in Figure 4, users interact with the external application through an auto-generated interface. In the screenshot of Figure 4, the user accesses an external service that selects ship crew that meet certain criteria. User input will be automatically converted to the expected API parameters of the external Cloud service. All user interface elements (labels and field values) in the above example are generated automatically from the user task model.

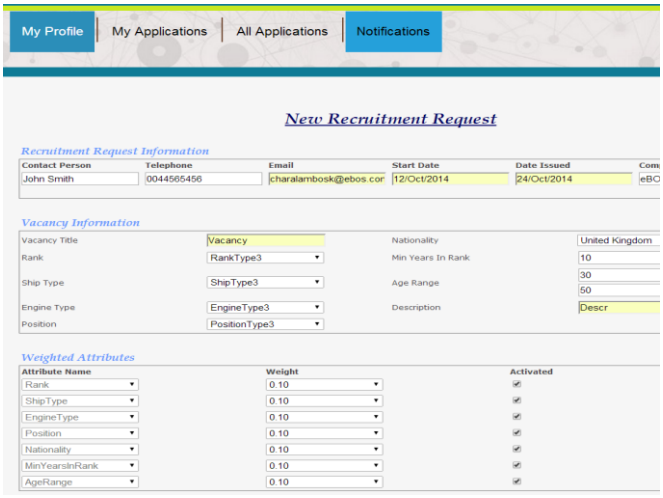


Figure 4: An end user application with a dynamically generated interface.

V. CONCLUSIONS AND FURTHER WORK

Shipping oriented portals are web-based communities which allow customers, intermediaries (e.g. forwarders) and carriers to communicate through a single portal for booking, tracking and tracing, documentation functions, etc. However, these portals offer a fixed set of functionalities and interfaces that assume specific user tasks. The approach described here presents several business and technical benefits, over them. With portal interfaces that adapt to user tasks, training costs can be significantly reduced, as the users do not have to learn the interface of several new applications. User satisfaction also improves, as the users feel comfortable in using applications whose interface matches the tasks they carry out in their domain of expertise. As new Cloud applications and services in a particular domain, emerge the users can access them seamlessly while maintaining a consistent interface.

However, there are also some restrictions in the current prototype and further areas of possible improvement. For start, a one to one mapping between a user task and a Cloud application/service is assumed. It is also assumed that the user tasks are independent from each other, while in practice these tasks are often joined in a user oriented workflow. It would be desirable to allow users to specify not only tasks but process workflows as perceived by them. Ideally, this would have to be done via a graphical environment. Integration of external cloud applications/services into this workflow would had to be carried out by the portal, as the Cloud applications are by definition application/process agnostic. Thus, the mediation manager discussed in this paper would also have to become a workflow manager.

The next area of improvement concerns the portal administrator's development environment. This environment is currently form based, where the administrators are given options to configure user profiles as well as the profiles of the external cloud services and applications. Generation of dynamic interfaces and messaging communications are handled by portal's internal services. A fully fledged development environment, would allow portal developers and administrators more flexibility in configuring access and integration with external Cloud services and further customisation of the end user options. This could be achieved by a system level scripting language such as those used by other Cloud management environments.

ACKNOWLEDGMENT

Research reported in this paper was partially supported by the EU under Project eMAR (Grant Number 265851 DG MOVE).

REFERENCES

[1] Yu, D. ; Jian W. ; Hu, B. ; Liu, J .; Zhang, X. ; He, K. ; Zhang, L-J. *A Practical Architecture of Cloudification of Legacy Applications*. IEEE World Congress on Services (SERVICES), 2011 pp 17-24.

[2] Zhang, Y. ; Wang, X. ; Su, GHL ; Wang, D. *Portable desktop applications based on user-level virtualization*. Computer Systems ACSAC 2008. 13th Asia-Pacific Architecture Conference, pp 1-6.

[3] Alrokayan, M. & Rajkumar B. *A web portal for management of aneka-based MultiCloud environments*. Proceeding AusPDC '13 Proceedings of the Eleventh Australasian Symposium on Parallel and Distributed Computing 2013- Volume 140, pp 49-56.

[4] Powell, C.; Munetomo, M. ; Wahib, A. ; Aizawa, T. *Constructing a Robust Services-Oriented Inter-cloud Portal Based on an Autonomic Model and FOSS* IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC), 2013, pp 458-463,

[5] Gmelch, O. *User-Centric Application Integration in Enterprise Portal Systems* . Josef Eul Verlag GmbH– 31 Jul 2012

[6] Hohpe, G. & Woolf, B. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*– Addison Wesley Signature October 20, 2003

.

Reverse Auction-based Resource Allocation Policy for Service Broker in Hybrid Cloud Environment

Sunghwan Moon, Jaekwon Kim,
Taeyoung Kim, Jongsik Lee
Department of Computer and Information Engineering, Inha University
Incheon, South Korea
e-mail: shmoon@inhaian.net, jaekwonkorea@naver.com,
silverwild@gmail.com, jslee@inha.ac.kr

***Abstract*— Hybrid cloud service utilizes public cloud and private cloud to provide its service. Furthermore, the hybrid cloud requires the resource allocation model to guarantee the Service Level Agreement (SLA), and minimize the cost. In this paper, we propose the Reverse Auction-based Resource Allocation Policy for Service Broker (RARAP). RARAP defines and utilizes the internal property of nodes on hybrid cloud environment. We simulate and evaluate the performance with the deadline compliance rate and the service usage cost. The simulation result proves the efficiency of our proposed model.**

Keywords*— *Resource Allocation Policy, Reverse Auction, Hybrid Cloud, RARAP.

I. INTRODUCTION

Globally, big data processing has become a major issue in various fields. Hence, Internet-based service is showing a tendency to rise. Therefore, a demand and importance of high-performance computing are also increasing continuously. Cloud computing utilizes the virtualization technique to construct the computing environment. It allows the cloud environment to provide high performance with distributed resources. In recent years, cloud computing has become an important part of business and industry [1].

Cloud computing is classified into three types of services. Software as a service (SaaS) aimed at providing the contents service for the user. Platform as a Service (PaaS) is concerned with processing for service requests. Infrastructure as a Service (IaaS) is interested in resource virtualization for job processing with physical resources. In addition, many services are under development for cloud computing [2].

Hybrid cloud provides data processing service using the public and private cloud. The public cloud is the paid service from external providers. On the other hand, the private cloud is the internal system with free service [3]. The collaboration with the public and private cloud may not only reduce the cost, but also increase the utilization. The service provider may also construct the resource depending on the cost. For this purpose, the system includes the service broker. The service broker automatically manages the cost to create an added value for both cloud service providers and users. This allows the hybrid cloud to minimize the cost, to utilize the various services, to manage the resource performance, and to provide the service [4]. However, hybrid cloud is vulnerable to an increasing number of service requests and complexity. Because of this, the hybrid cloud hardly provides the Service Level Agreement (SLA) for service providers and users [1]. Hence, hybrid cloud requires a new SLA-guaranteed method that minimizes cost.

In this paper, we propose the reverse auction-based resource allocation policy for service broker (RARAP) on hybrid cloud environment. RARAP defines a cost and an internal property of resources for processing a job by a deadline. RARAP utilizes the reverse auction to estimate the processing cost and allocation priority [5]. In other words, the reverse auction is to approximate the service usage cost for a resource on the hybrid cloud. Then, RARAP assigns the job to the most suitable resource using the reverse auction. RARAP ensures the SLA at a low cost in a hybrid cloud. The proposed method may be utilized for defense modeling and simulation. Battlefield data requires a large amount of computation resource to get meaningful results. Thereby, the cloud-based approach is the best choice for battlefield data analysis. And the reverse auction ensures a high efficiency for resource allocation with a reasonable cost.

The rest of this paper is structured as follows: In section 2, we briefly review the related works. Section 3 describes our key idea for cloud resource allocation policy. Section 4 explains the simulation design and results. Finally, we conclude in Section 5.

II. RELATED WORKS

A. Business Model for Resource Management on Cloud Computing

Up to now, much study has been done in the business model and job scheduling technique for cloud computing environment.

A commodity market model [6] has been proposed to connect between service providers and service users. The service provider fixes the resource fee and the parameter, which is based on the service users' usage. The commodity

 ISBN: 978-1-61208-388-9

market model has the fixed price policy for resource providing. The user does not participate in the price fixing.

An auction model [7] is most generally used in the parallel and distributed computing environment. Both service providers and service users tender the service condition. The auction model selects the service provider who suggests the most suitable condition for users' demand. Hence, the auction model shows the asymmetric feature for price fixing.

B. Cloud Resource Management and Scheduling

Resource management and scheduling is also one of the most studied topics on the cloud and distributed computing.

The cloud service often receives a complex application request from the user. The hybrid cloud utilizes the public cloud to comply with the service deadline. Because of this, Van den Bossche et al. [3] proposed a scheduling method with the cost minimizing technique. However, the cost minimizing method only considers the service cost for scheduling. This feature assigns more jobs to the free cloud service. As a result, the cost minimizing method causes a bottleneck problem on the private cloud. Hence, the variable deadline may affect the failure rate.

The ontology-based management is based on the semantic and prediction approach. The ontology-based system constructs the resource candidates with the user's requirement. The system selects the most suitable method from all the candidates to comply with the SLA [8].

This paper aims to reduce the cost and satisfy the SLA. For this, we consider the cost, performance, job size, and deadline with the model based reverse auction.

III. REVERSE AUCTION-BASED RESOURCE ALLOCTAION POLICY

We propose RARAP to minimize the cost and ensure SLA compliance. RARAP uses the reverse auction method based on the internal property of the resource. RARAP also performs the re-scheduling technique to improve the throughput. Through this, RARAP minimizes the cost for hybrid cloud services. Figure 1 shows the architecture of RARAP.

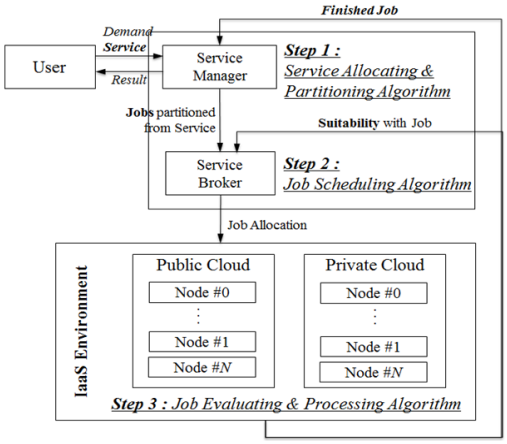


Figure 1. Architecture of RARAP

RARAP performs the procedure in a five step for resource allocation. These phases perform as follows:

A. Service request and divided into job

User is sequentially requesting services. Service manager divides the received service request in a number of jobs. For example, we assume that user requests a service provided by the save and preview the image file. Service manager divides the service into works of uploading an image file to the server, securing a storage space for image files, and creating a thumbnail for the preview. Service has the size and deadline as internal properties. Internal properties of the job follow those of the service. However, the job size is divided by the size of the service in terms of a number of jobs.

B. Delivery and classification of job

Service manager sequentially sends the job to Service broker. Service broker stores the incoming job from Service manager in the queue with the consideration of the size and the deadline. Service broker has a circular queue, and linear queues as many as the number of nodes. The job transmitted from Service broker is stored in the circular queue and waits for calculating the job suitability.

C. Job evaluation anad suitability calculation

If Service broker posts the job to be processed, all the nodes in the cloud environment return the job suitability. The job suitability is a score indicating the node efficiency of the job processing. That is, all the nodes follow the reverse auction method of competition through their performance. The score is sent to Service broker again to determine the node for job assigning. The job assigned to the node is stored in a linear queue.

D. Job processing and updating of the nodes

All nodes in the hybrid cloud are waiting to receive the job from Service broker. The linear queue of Service broker is a job queue for each node. The broker delivers the waiting job on the queue to the node, when the node is empty. The node processes the received job, and then updates to ready state in order to process the next job. The node also sends the finished job to Service manager.

E. Job merging and returning serivce

The finished job waits on Service manager for merging. When all jobs belonging to the same service have arrived, the jobs are merged into the service. The merged service is presented to the requesting user.

IV. SIMULATION DESIGN AND RESULTS

We design the hybrid cloud environments to test the effectiveness of our proposed RARAP. The environment is based on the Discrete Event System Specification (DEVS)

 ISBN: 978-1-61208-388-9

formalism [9], and measures the usage cost and the deadline compliance rate.

A. Simulation Design

We design the simulation model based on Figure 1. This simulation is designed to demonstrate the following effects of the RARAP in small hybrid cloud environment. The first is to ensure the SLA with the compliance of the job deadline. The last is the cost reduction for the same service requirements.

User sends the service request to Service Manager. Service manager divides the service into jobs, and distributes the divided jobs to the Service broker. Every node calculates the job suitability score, and returns the result to the Service Broker. Then, the Service broker finally assigns the job to the specific node. The node processes the assigned job from the Service broker. The solved job is transmitted from the node to the Service manager. Then, the Service manager merges the jobs into the service, and returns it to the user.

The hybrid cloud utilizes both the private and public nodes for service. We define the performance of both nodes for simulation as shown in Table I. The processing speed in Table I is in exact proportion to the processing time. The lower value for processing speed indicates the less time for job processing.

TABLE I. SIMULATION CONFIGURATION – NODE PERFORMANCE

Node Number	Node Type	Processing Speed	Usage Cost
0	Private	5.5	0
1	Private	10.5	0
2	Private	2.6	0
3	Public	1.1	7
4	Public	1.7	5
5	Public	2.7	2

The public node takes a service usage cost to provide the public cloud service such as Amazon Web Service [10] or Window Azure [11]. On the other hand, the private node refers to the SOHO server and network attached storage that may be held by individuals or small companies. Table II shows the usage cost policy for public node on our simulation. This pricing policy is defined on a scale from 0 to 10 according to the CPU performance, which is offered by the public cloud service. The price is based on the size and the deadline used in the service parameter for experimental environment.

TABLE II. SIMULATION CONFIGURATION – PUBLIC CLOUD USAGE COST

Usage Cost	1	2	3	4	5	6	7	8	9	10
Processing Speed	0.2	0.5	0.8	1.1	1.4	1.7	2.0	2.3	2.6	2.9

As mentioned above, each node calculates the job suitability score. The estimation result is based on the processing speed and cost. Table III shows score tables for each factor.

TABLE III. SCORE FOR CALCULATING SUITABILITY

Speed Score	1	2	3	4	5
Processing Speed	22.8 ~ 25.0	20.6 ~ 22.8	18.4 ~ 20.6	16.2 ~ 18.4	14.0 ~ 16.2
Speed Score	6	7	8	9	10
Processing Speed	11.8 ~ 14.0	9.6 ~ 11.8	7.4 ~ 9.6	5.2 ~ 7.4	3.0 ~ 5.2
Speed Score	11	12	13	14	15
Processing Speed	2.6 ~ 2.9	2.3 ~ 2.6	2.0 ~ 2.3	1.7 ~ 2.0	1.4 ~ 1.7
Speed Score	16	17	18	19	20
Processing Speed	1.1 ~ 1.4	0.8 ~ 1.1	0.5 ~ 0.8	0.2 ~ 0.5	0.0 ~ 0.2

CostScore	1	2	3	4	5	6	7	8	9	10
Cost	10	9	8	7	6	5	4	3	2	1
Processing Speed	0.2	0.5	0.8	1.1	1.4	1.7	2.0	2.3	2.6	2.9

SpeedScore in Table III is defined on a scale from 0 to 20 according to the performance of all the developed CPU for a personal computer [12]. We assume that the performance of CPU, which is held by nodes in the cloud, is proportional to the processing speed. However, since the public cloud is a paid service, the suitability calculation considers this feature for grading the nodes. Each node converts its own performance information to SpeedScore and CostScore by using Table III. The node calculates the job suitability using (1).

```
If(Node of Private Cloud)
Job Suitability = SpeedScore

If(Node of Public Cloud)
Job Suitability = SpeedScore - CostScore
```

(1)

In our simulation, the user requests the services from a minimum of 50 up to 500. Both size and deadline of service are based on the Wikipedia Page Traffic V3 Statistic [13], which is opened through the public data sets of Amazon Web Service. We use this public data to the processing to meet the needs of our environment.

We measure the service performance with three different models.

- First, we use the sequentially assigned model for cloud service, "round-robin". The round-robin model sequentially assigns jobs to all nodes. In other words, the job is assigned in the order of nodes, regardless

of the performance indices. Therefore, the service is returned in the order requested from the user.

- Second, we use the randomly assigned model with a table of random numbers. The random model distributes the job on the basis of the calculated suitability. However, this model will randomly select a node from the candidate group.
- Last is our proposed method RARAP. RARAP allocates the job to the node having the highest goodness of suitability as mentioned above.

B. *Simulation Results*

We measure the deadline compliance rate and the total usage cost for each comparison model. The purpose of this experiment is to verify the proposed RARAP can guarantee the SLA at an affordable cost.

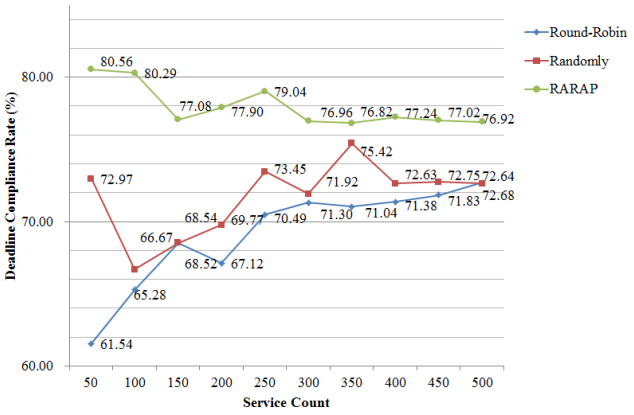


Figure 2. Result of Graph for Deadline Compliance Rate

Figure 2 shows the measured result for deadline compliance rate. As presented in (2), the deadline compliance rate is the percentage of solved jobs before the deadline. It may show the processing efficiency of each model.

$$\text{DeadlineComplianceRate}(\%) = \frac{\text{DeadlineComplianceJob}}{\text{TotalCompletedJob}} \quad (2)$$

As shown in Figure 2, the round-robin model records a deadline compliance rate of 69.118%, the random model records 71.676%, and our proposed RARAP records 77.983%. This value is an average percentage of the deadline compliance result. Our proposed RARAP considers the deadline to assign the job. As a result, our model shows superior compliance rate and less variation than other models.

Figure 3 shows the other measured result for processing cost. This result is to present the price effectiveness for each model. We measure the processing cost in the same throughput for fair comparison.

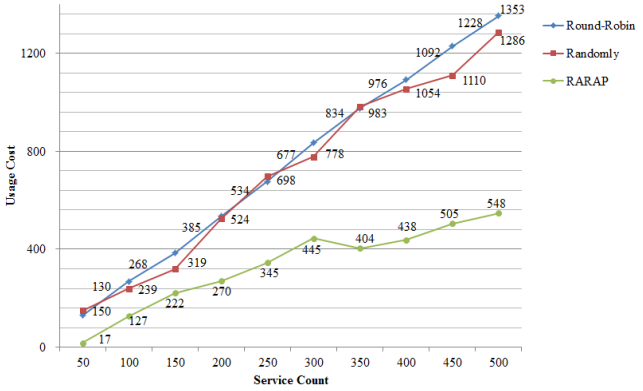


Figure 3. Result of Graph for Usage Cost

As shown in Figure 3, the round-robin model records 747.70, the random model records 714.10, and our proposed model RARAP records 332.10. In our design, only the public node charges the service usage cost with the price policy shown in Table III. Our model tries to minimize the processing cost. RARAP avoids the public node under the same conditions. The public node is inevitable choice for our model. This method induces the least processing cost for RARAP.

V. CONCLUSION AND FUTURE WORK

This paper proposes reverse auction-based resource allocation policy for service broker in hybrid cloud environment. RARAP utilizes the reverse auction method, and assigns the resource with three steps. RARAP controls the job schedule with the job suitability score, which is based on the processing speed and service usage cost. It may improve the efficiency, and decrease the cost and the number of SLA violations.

Future work will concentrate on the data partitioning. The interval-based partitioning management can increase the utilization per cost for cloud resources. Our study will be used in the analysis of battlefield data. The defense simulation has traditionally required a large amount of processing resources. The proposed model is expected to be able to meet the analysis data required for war game simulation.

ACKNOWLEDGMENT

This research was supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract UD140022PD, Korea, and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012R1A1A2002751).

REFERENCES

[1] J. Koh, H. Kang, and Y. Kim, “Adaptive policy-based task scheduling for scientific applications in hybrid cloud,” Journal

of KIISE: Computing Practices and Letters, 19th vol.11, 2013, pp. 572-579.

[2] J. Kim and J. Lee, "Fuzzy logic-driven virtual machine resource evaluation method for cloud provisioning service," Journal of The Korea Society for Simulation, 22nd vol.1, 2013, pp. 77-86.

[3] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads," Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference, 2010, pp. 228-235.

[4] J. Kim, D. Kang, N. Kim, J. Lee, and S. Jung, "Cloud service broker managing and integrating multiple heterogeneous clouds," Communications of KIISE, 32nd vol.2, 2014, pp. 52-58.

[5] S. D. Jap, "The impact of online reverse auction design on buyer-supplier relationships," Journal of Marketing, 71st vol.1, 2007, pp. 146-159.

[6] Y. Amir, B. Awerbuch, A. Barak, R. S. Borgstrom, and A. Keren, "An opportunity cost approach for job assignment in a scalable computing cluster," Parallel and Distributed Systems, IEEE Transactions, 11st vol.7, 2000, pp. 760-768.

[7] M. Stonebraker et al., "An economic paradigm for query processing and data migration in Mariposa," Parallel and Distributed Information Systems, 1994., Proceedings of the Third International Conference, 1994, pp. 58-67.

[8] Y. B. Ma, S. H. Jang, and J. S. Lee, "Ontology-based resource management for cloud computing," Intelligent Information and Database Systems, Springer Berlin Heidelberg, 2011, pp. 343-352.

[9] B. P. Zeigler, H. Praehofer, and T. G. Kim, "Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems," Academic Press, 2000, pp. 76-96.

[10] Amazon Web Service. [Online]. Available from: http://aws.amazon.com/ 2013.11.01

[11] Windows Azure. [Online]. Available from: http://www.windowsazure.com/ 2013.11.01

[12] CPU Benchmarks by PassMark Software. [Online]. Available from: http://www.cpubenchmark.net/cpu_list.php 2014.03.04

[13] Wikipedia Page Traffic V3 Statistic. [Online]. Available from: https://aws.amazon.com/datasets/6025882142118545 2013.10.05

 ISBN: 978-1-61208-388-9

Deployment of Virtual InfiniBand Clusters with Multi-tenancy for Cloud Computing

Viktor Mauch

Steinbuch Centre for Computing (SCC)
Karlsruhe Institute of Technology (KIT)
Email: viktor.mauch@kit.edu

***Abstract*—Today, most high performance computing (HPC) systems are equipped with high-speed interconnects providing low communication and synchronization latencies in order to run tightly coupled parallel computing jobs. They are typically managed and operated by individual institutions and offer a fixed capacity and static runtime environment with a limited selection of applications, libraries and system software components. On the contrary, a cloud-based Infrastructure-as-a-Service (IaaS) model for HPC resources promises more flexibility, as it enables elastic on-demand provisioning of virtual clusters and allows users to modify the runtime environment down to the operating system level. The goal of this research effort is the general demonstration of a prototypic HPC IaaS system allowing automated provisioning of virtualized HPC resources while retaining high and predictable performance. We present an approach to use high-speed cluster interconnects like InfiniBand within an IaaS environment. Our prototypic system is based on the cloud computing framework Openstack in combination with the Single Root - I/O Virtualization (SR-IOV) mechanism for PCI device virtualization. Our evaluation shows that, with this approach, we can successfully provide dynamically isolated partitions consisting of multiple virtual machines connected over virtualized InfiniBand devices. Users are put in the position to request their own virtualized HPC cluster on demand. They are able to extend or shrink the assigned infrastructure and to change the runtime environment according to their needs. To ensure the suitability for HPC applications, we evaluate the performance of a virtualized cluster compared to a physical environment by running latency and High-Performance Linpack (HPL) benchmarks.**

Keywords–HPC, InfiniBand, Cloud Computing, Virtualization, Openstack

I. INTRODUCTION

In recent years, cloud computing [1][2] has influenced significantly most parts of information technology. The consumption of applications, services and infrastructure provided by public operators, has increased dramatically. However, still today the demand of High Performance Computing (HPC) resources is typically covered by local installations provided and used by single institutions. Such physically operated clusters have disadvantages. Due to specific requirements regarding performance and scope, it is common to deploy a predefined, fixed runtime environment with specific applications, libraries, job schedulers and operating systems. As a result, users are limited to implement customized application scenarios based on modifications of the underlying operating system or other important core runtime libraries. Furthermore, demand is fluctuating, resulting in periods where physical resources are underutilized or overloaded.

A High Performance Cloud Computing (HPC2) [3] model based on an Infrastructure-as-a-Service (IaaS) delivery solution promises more flexibility and efficiency in terms of cost and energy consumption. It allows moving away from physically owned but underutilized HPC clusters designed for peak workloads to vitualized elastic HPC resources leased from a consolidated large HPC computing center working near full capacity. The deployment of virtual machines (VM) allows users to securely gain administrative privileges and customize the runtime environment according to their specific demands. Incurred costs are associated directly by a *pay-as-you-go* model with the corresponding resource usage or with the responsible user respectively.

In the next Section, we discuss challenges concerning HPC in the cloud, followed by the architecture description in Section III. Section IV includes some detailed information about our prototypic implementation. Based on that, a performance evalution is provided in Section V. Conclusion and outlook can be found in Sections VI and VII.

II. HPC CLOUD CHALLENGES

Providing cloud-based HPC services raises difficult challenges. Virtualization, the core technique of general purpose IaaS offerings, certainly achieves the desired elasticity and multi-tenancy. On the other hand, virtualized environments are associated with a higher overhead and may lead to unpredictable variations in performance. Early studies [4][5] concerning the evaluation of the Elastic Compute Cloud (EC2) standard services, provided by the Amazon Web Services (AWS), confirm these observations. Further research work [6][7] concerning the execution of HPC applications in contemporary non-HPC IaaS environments has identified the network performance as the primary hindrance to implement virtualized HPC clusters. Therefore, I/O virtualization is one of the key challenges of providing HPC cloud resources with high-speed interconnect support. Since 2010, Amazon provides so-called *cluster compute* instances for EC2, which are equipped with a 10GbE interconnect and thus are more capable to handle typical HPC tasks with an acceptable performance [8]. In theory, this service allows to build up a virtualized HPC cluster listed in the TOP500 list, which was demonstrated by Amazon for advertising purposes. Over 1064 instances with 17024 cores reached place 42 of the TOP500 list (November

 ISBN: 978-1-61208-388-9

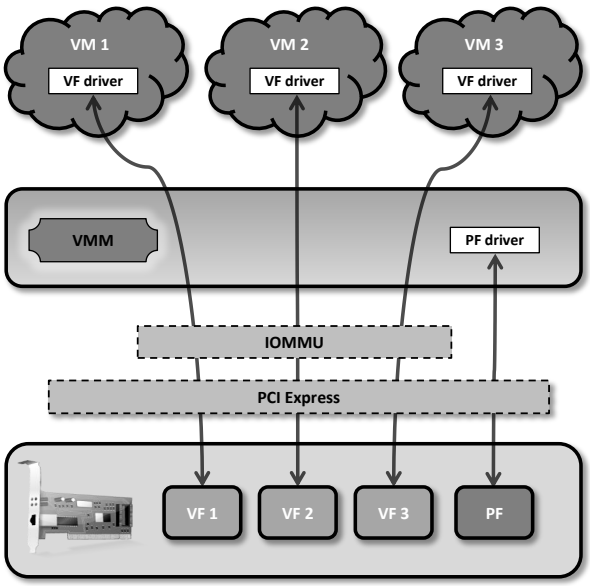


Figure 1. Virtualized I/O device access via SR-IOV. The PCI device presents itself as several *virtual functions* which are allocated to the VMs via PCI passthrough.

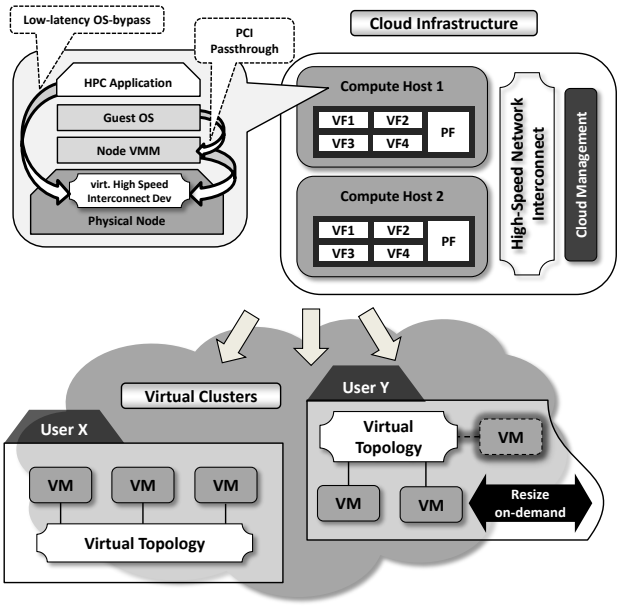


Figure 2. Virtualization of host and the cluster interconnect is managed by a cloud computing framework to provide elastic virtual servers.

2011) with an HPL benchmark of 240TFlops/s. However, the provisioning of VMs with a high-speed interconnect, such as InfiniBand improves performance significantly [9].

InfiniBand (IB) [10] has a substantial performance advantage due to the processing of all network layers within the device hardware. Especially, tightly coupled HPC applications benefit from the very low communication latency in comparison to a traditional network technology such as Ethernet. However, using IB within a virtualized environment is a non-trivial task that can only be partially achieved by software-based approaches. Li et al. [11] have proposed Virtual Machine Monitor (VMM)-bypass I/O, a para-virtualization approach for InfiniBand on Xen. This solution requires ongoing modifications of drivers in host and guest with respect to changes of the underlying hardware and operating system. The concept of Peripheral Component Interconnect (PCI) passthrough grants a VM direct and exclusive access to a dedicated PCI I/O device. It requires an I/O Memory Mapping Unit (IOMMU) to ensure memory protection between different VMs [12] and restricts the number of VMs per host to the number of I/O devices built in. A more suitable solution would be *Single Root - I/O Virtualization (SR-IOV)* [13], which allows a single PCI Express device to appear as multiple, separate devices, called *Virtual Functions (VF)*, a kind of a "light weighted" PCIe function. Each VM can be allocated to one VF via PCI passthrough. The *Physical Function (PF)* includes the SR-IOV capability and has full configuration resources such as discovery, management and manipulation. It is an anchor for creating VFs and reporting errors and events. Figure 1 provides an overview of the SR-IOV dependencies.

In this paper, we present an architecture for the deployment of multi-tenant virtual clusters, based on the virtualization of the IB interconnect, with acceptable performance and latencies compared to native clusters. In principle, the following implementation design is based on our former research work [3][14]. A detailed evaluation of the architectural approach based on PCI passthrough in combination with OpenNebula has also been worked out by Hillenbrand [15]. With this paper, we extend our approach by using SR-IOV and the deployment of multiple VMs per hosts with IB support within a cloud computing IaaS environment. The next sections provide a fundamental description of the architecture and the prototypic implementation of our solution followed by a basic performance evaluation.

III. ARCHITECTURE

In relation to the National Institute of Standards and Technology (NIST) definition [1] of cloud computing, two essential characteristics are crucial for HPC IaaS cloud systems. *Resource Pooling* and infrastructure multitenancy is necessary to provide cloud computing services to multiple independent users. It demands the isolation of each cluster network at any time. The user gets the impression to use the interconnect exclusively, albeit with reduced bandwidth. *On-demand self-service* requires automated allocation of virtual HPC resources including the configuration of the cluster network interconnect. Furthermore, *service level agreements* on the minimum network quality have to be guaranteed as HPC tasks may heavily depend on it. Figure 2 illustrates our architectural approach. We extend an existing cloud computing framework with features to manage the provisioning of virtual clusters as well as the configuration of the underlying IB interconnect topology. Available *Virtual Functions* are allocated to new VMs which are able to run HPC applications using the virtual IB device by low-latency bypass.

Using SR-IOV for IB virtualization simplifies many aspects with respect to network isolation, management and security. A VF assigned to the VM is restricted by the physical device hardware compared with an exclusive access to a dedicated

 ISBN: 978-1-61208-388-9

physical device via PCI passthrough. First of all, users with administrative privileges within a VM are not able to modify the firmware of the physical PCI device anymore. This is a very important aspect for cloud infrastructure, which is used by multiple users over time. Further restrictions prevent the execution of a *subnet manager* within a VM, which could be used to reconfigure the whole IB network topology.

Users should be able to provide their own VM templates with the corresponding software environment and deploy and resize their VM ensembles on-demand. The underlying cloud computing framework must ensure the network isolation of each user specific VM ensemble at any time. At first glance, the provision of multiple VMs with high performance network interconnect support on single hosts seems to be impractical concerning the usage of HPC workloads. Users would always try to allocate the available hardware with few overhead as possible. However, mixed IaaS environments with common and HPC-capable VMs could utilize the available infrastructure more efficiently. Furthermore, resource over-provisioning could also reduce operation costs. While users would get lower guarantees concerning computing and networking quality, IaaS advantages like flexibility and on-demand provision still can be used compared with the traditional operation of native HPC clusters.

IV. Implementation

The orchestration of HPC resources and the configuration of the network infrastructure with respect to isolated partition is done by a cloud computing framework. We decided to use the Openstack framework, which is currently one of the most popular and promising open-source cloud computing IaaS frameworks on the market. The latest stable version with the codename *Havana*, which we use for our prototypic implementation, already provides the necessary PCI passthrough mechanisms for the assignment of PCI devices to VMs. Openstack supports several hypervisor solutions. We decided to use the Kernel-based Virtual Machine (KVM) hypervisor for node virtualization running on Linux, as Linux can be seen as the de-facto standard operating system for HPC systems [16] and is supported by Mellanox with SR-IOV capable drivers.

The underlying hardware infrastructure to operate our prototypic HPC cloud consists of two *Dell R710* servers. Each of them is equipped with two *Intel Xeon E5620* quadcore 2.66GHz cpus, 64 GB of RAM and a *Mellanox ConnectX-2 InfiniBand Quad Data Rate (QDR) HCA*, which is configured to provide 7 VFs to the host system. Both nodes are connected with Ethernet (1 Gbit/s) and an InfiniBand Doube Data Rate (DDR) switch, which provides a bandwidth of 20Gbit/s. *CentOS 6.4* is used as host and guest operating system together with the *Mellanox OFED 2.0* software stack, which provides drivers and management tools for the integrated IB devices.

To ensure and manage the isolation of virtual clusters with IB partitions at any time, we extend the Openstack framework with the necessary functionality. The IB subnet manager provides a pre-configured *partition key (pkey)*-table to all existing Openstack compute nodes with IB devices. Our Openstack extension registers continuously any changes concerning all available VMs and their associated users. The isolation of an ensemble of logically related VMs is accomplished by assigning a specific pkey to the corresponding VFs. This is done by automated configuring the virtual-to-physical pkey mappings within the host operating systems of the compute nodes. Thereby each virtual IB cluster gets its own pkey and is blocked communicating to other virtual clusters over IB or manipulating IB network topology. This mechanism has similarities with respect to the Virtual Local Area Network (VLAN) technique used by Ethernet network technology. So, our prototypic OpenStack implementation allows users to deploy and resize their VM cluster without having network conflicts with other VMs in the same IB subnet.

Using SR-IOV leads to limitations. Although the SR-IOV specification allows up to 255 VFs per PCI device, the actual usable number is often extremely lower (7–15) because of strong dependencies on the BIOS, chip set and adapter hardware. The capacity of the pkey-table also depends on the provided hardware and is more significant. The above mentioned *Mellanox HCAs* used in our prototypic system are limited to 128 *pkeys*. These circumstances must be taken into account concerning a possible *scale up/out* of the cloud infrastructure.

TABLE I. HIGH PERFORMANCE LINPACK BENCHMARK

Infrastructure	vCPU config	GFlops	Efficiency
2 nodes (8 Cores)	N/A	156.8	92.1%
4VMs (4 vCores)	dynamic	149.4	87.8%
4VMs (4 vCores)	fixed	152.8	89.8%
8VMs (2 vCores)	dynamic	141.3	83.0%
8VMs (2 vCores)	fixed	143.5	84.3%

V. Performance Evaluation

In this section, we present a basic performance evaluation of our early prototypic system. In order to get an impression of the HPC performance, the High Performance Linpack (HPL) [17] benchmark has been executed on several virtual cluster scenarios as well as on the underlying native hardware, see Table I. Therefore, we have used *Intel Optimized LINPACK Benchmark*, which is based on Intel Math Kernel Library (MKL) and the Intel MPI implementation. The corresponding *HPL.dat* tuning parameters for all test scenarios are set as follows: N=100k, NB=168, p=4, q=4. The peak performance of our *Westmere*-based 16-core infrastrucutre is calculated to R_{max}=170.2 Gflops. The test results, we obtained, are looking very promising. As expected, result values provided by virtual clusters are weaker compared to the native environment. When multiple VMs with less virtual cores are combined into virtual clusters, additional virtualization / communication latencies might play a role. On the other hand, an efficiency above 80% is quite noteworthy, considering that native HPC clusters based on Ethernet network technology barely reach an efficiency of 70%. Worth mentioning is also a slight increase in efficiency by a corresponding virtual Central Processing Unit (vCPU) configuration, which ensures that each virtual core is assigned permanently to a fixed physical core.

Especially MPI applications strongly depend on low communication latency. We have run SKaMPI [18], a synthetic MPI benchmark, between two VMs on the same physical node as well as on distributed nodes. In addition, we have performed the same measurement on the native hardware. VMs on the same host are able to communicate with QDR speed directly through the attached *ConnectX-2 HCA*. However, the communication between both hosts is downgraded to DDR speed because of the connection over an IB DDR

 ISBN: 978-1-61208-388-9

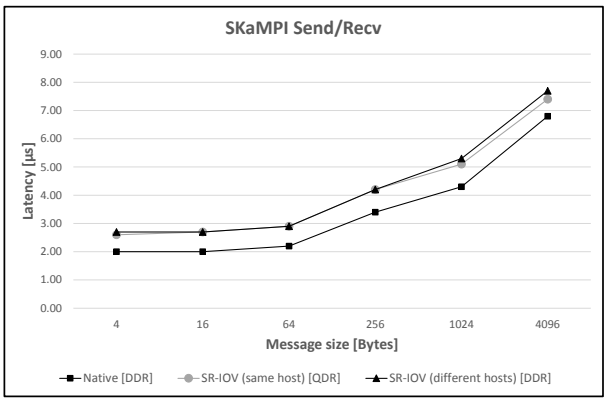


Figure 3. Measured communication latencies in μs.

switch. Figure 3 presents the results. It turns out that the communication latency within virtual cluster increases only slightly compared with the native environment. In summary, it can therefore be said that our prototypic HPC system has the necessary prerequisites to provide an acceptable environment for executing HPC tasks. A more detailed evaluation for SR-IOV performance in combination with the IB interconnect is done by Panda et al. [19].

VI. Conclusion

An IaaS model for HPC based on cloud computing allows users to request elastic virtual HPC clusters as on-demand resources. Compared to native environments, virtual resources are provided to users with administrative privileges and billed according to the *pay-as-you-go* principle. We adapt this architecture model for the operation of an HPC cloud based on the Openstack framework and the IB cluster interconnect. Therefore, we use the SR-IOV specification for PCI devices to manage and utilize the available HPC infrastructure more efficiently. Users get independent isolated clusters with better Linpack performance efficiency and communication latency compared to virtualized and native cluster infrastructure based on Ethernet.

VII. Outlook

Our next steps include extending the infrastructure test bed and extensive testing of typical HPC applications to provide a more detailed evaluation of performance impacts within virtualized HPC environments. Furthermore, we are looking forward to compare our findings with the announced HPC IaaS resources, which will be provided by Microsoft Windows Azure this year. New compute intensive VM instances of type A8/A9 will also support the IB interconnect for running HPC tasks. However, at the moment we have no information about the specific deployment technology.

Currently, live migration is an important mechanism for cloud computing services as it eases cluster management and allows load balancing and fault tolerance. But it is still challenging to achieve with high-speed network interconnects like IB, as these technologies usually maintain their connection state within the network device hardware. Tasoulas [20] presented a first prototypic implementation of live migration over SR-IOV enabled IB devices. We will pursue this research field and try to adjust our architecture if possible.

References

[1] P. Mell and T. Grance, "The nist definition of cloud computing," National Institute of Standards and Technology, vol. 53, no. 6, 2009, p. 50.

[2] C. Baun, M. Kunze, J. Nimis, and S. Tai, "Cloud computing: Web-based dynamic it services," 2011.

[3] V. Mauch, M. Kunze, and M. Hillenbrand, "High performance cloud computing," Future Generation Computer Systems, vol. 29, no. 6, 2013, pp. 1408–1416.

[4] J. Napper and P. Bientinesi, "Can cloud computing reach the top500?" in Proceedings of the combined workshops on UnConventional high performance computing workshop plus memory access workshop. ACM, 2009, pp. 17–20.

[5] G. Wang and T. E. Ng, "The impact of virtualization on network performance of amazon ec2 data center," in INFOCOM, 2010 Proceedings IEEE. IEEE, 2010, pp. 1–9.

[6] C. Evangelinos and C. Hill, "Cloud computing for parallel scientific hpc applications: Feasibility of running coupled atmosphere-ocean climate models on amazon's ec2," ratio, vol. 2, no. 2.40, 2008, pp. 2–34.

[7] A. Gupta and D. Milojicic, "Evaluation of hpc applications on cloud," in Open Cirrus Summit (OCS), 2011 Sixth. IEEE, 2011, pp. 22–26.

[8] P. Zaspel and M. Griebel, "Massively parallel fluid simulations on amazon's hpc cloud," in Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on. IEEE, 2011, pp. 73–78.

[9] N. Regola and J.-C. Ducom, "Recommendations for virtualization technologies in high performance computing," in Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. IEEE, 2010, pp. 409–416.

[10] InfiniBand Architecture Specification Volume 1, Release 1.2.1. InfiniBand Trade Association, 2007.

[11] J. Liu, W. Huang, B. Abali, and D. K. Panda, "High performance vmm-bypass i/o in virtual machines." in USENIX Annual Technical Conference, General Track, 2006, pp. 29–42.

[12] B.-A. Yassour, M. Ben-Yehuda, and O. Wasserman, "Direct device assignment for untrusted fully-virtualized virtual machines," Tech. rep., IBM Research Report H-0263, Tech. Rep., 2008.

[13] P. SIG, "Single root i/o virtualization and sharing specification, revision 1.0," 2008.

[14] M. Hillenbrand, V. Mauch, J. Stoess, K. Miller, and F. Bellosa, "Virtual infiniband clusters for hpc clouds," in Proceedings of the 2nd International Workshop on Cloud Computing Platforms. ACM, 2012, p. 9.

[15] M. Hillenbrand. Towards virtual infiniband clusters with network and performance isolation. System Architecture Group, Karlsruhe Institute of Technology (KIT), Germany. Last Access: February, 2015. [Online]. Available: http://os.ibds.kit.edu/ [retrieved: June, 2011]

[16] H. W. Meuer, "The top500 project. looking back over 15 years of supercomputing experience," PIK-Praxis der Informationsverarbeitung und Kommunikation, vol. 31, no. 2, 2008, pp. 122–132.

[17] J. Dongarra, P. Luszczek, and A. Petitet, "The linpack benchmark: past, present and future," Concurrency and Computation Practice and Experience, vol. 15, no. 9, 2003, pp. 803–820.

[18] R. Reussner, P. Sanders, L. Prechelt, and M. Müller, "Skampi: A detailed, accurate mpi benchmark," in Recent advances in parallel virtual machine and message passing interface. Springer, 1998, pp. 52–59.

[19] J. Jose, M. Li, X. Lu, K. C. Kandalla, M. D. Arnold, and D. K. Panda, "Sr-iov support for virtualization on infiniband clusters: Early experience," in Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on. IEEE, 2013, pp. 385–392.

[20] V. Tasoulas. Prototyping live migration with sr-iov supported infiniband hcas. HPC Advisory Council 2013. Last Access: February, 2015. [Online]. Available: http://www.hpcadvisorycouncil.com/events/2013/Spain-Workshop/pdf/4_Simula.pdf [retrieved: September, 2013]

 ISBN: 978-1-61208-388-9

A Context-aware, Intelligent and Flexible Ambient Assisted Living Platform Architecture

Hendrik Kuijs, Carina Rosencrantz, Christoph Reich
Faculty of Computer Science
Furtwangen University of Applied Science
Furtwangen, Germany
Email: {Hendrik.Kuijs, Carina.Rosencrantz, Christoph.Reich}@hs-furtwangen.de

***Abstract*—Ambient Assisted Living (AAL) solutions support older people in remaining longer in their own environment. For a system to be successful on the market, it is essential to be flexible and adaptable to the individual needs of the elderly person. In this paper, we present an approach for a context-aware, intelligent and flexible AAL platform architecture, that integrates existing concepts for home automation environments with an extendable platform for information, communication and learning to assist elderly users in their daily life. The platform has the intelligence to react on environmental changes, by including data provided by sensors or external services, as well as changes of the medical state of the user by using a person centered ontology to deliver adapted services at any time. The intelligence to do this is assured by lightweight and autonomous software agents. The custom platform itself is realized as a Platform as a Service (PaaS) in the cloud. The setup is a Private Cloud, that shares central services in the Public Cloud for better flexibility, scalability and maintainability. It features a PaaS management system to customize and preconfigure different environmental settings. The user is able to add new services on demand or adjust the configuration of the platform to his needs.**

Keywords*—*PaaS; AAL; Cloud; OSGi; software agents; context-aware.

I. INTRODUCTION

Due to the demographic change towards an aging population and the emerging shortage of care facilities, the field of AAL aims to support elderly people in their daily living to enable them to stay in their own homes as long as possible.

As part of the research project ZAFH AAL [1], the platform named Person Centered Environment for Information, Communication and Learning (PCEICL) has been designed [2]. PCEICL is a personal assistance system with the primary goal to assist elderly people in staying at home longer and in improving social participation in rural regions by delivering social services. An essential requirement for such a platform is an automatic adaptation and tailoring of the services to an elderly maybe handicapped person's needs.

In the field of AAL, Open Services Gateway initiative (OSGi) [3] is often used to provide an easy integration framework for sensors and actors in home-automation environments [4][5][6]. OSGi enables developers to build up modular systems or to reuse existing services and therefore supports the upgradeability and extensibility of AAL systems [7].

The key feature of PCEICL is the adaptation of functionality and presentation of information based on the medical state and the physical environment of the user. Information about the user is stored and retrieved by implementing the PCEICL ontology [8] and information about the environment is provided by attached sensors or external web-services. The platform makes intelligent decisions based on the provided information about the user and its environment. The PCEICL platform tackles the problem of intelligent decision making by a software agent platform, in this case Java Agent Development Framework (JADE) [9]. While other platforms have a strong focus on emergency detection and prevention, the main concept of PCEICL is to provide information and communication services to support social participation.

The outline of this paper is as follows: After presenting related work in Section II, Section III explains two different concepts for combining the OSGi component middleware and the software agent framework JADE. The architectural approach of the PCEICL platform with its different layers of agents, OSGi bundles and the PCEICL ontology is explained in Section IV and is put in relation to the cloud infrastructure in Section V. The flexibility, context awareness and intelligence of this architectural approach are worked out in Section VI. Section VII presents a first evaluation based on different scenarios, followed by a conclusion and a further outlook of upcoming research topics in Section VIII.

II. RELATED WORK

OSGi gained wide currency in the field of home automation and smart home as a middleware for different sensors and actors [3]. ProSyst [10] offers a *Home Gateway Middleware*, a *SDK* for developers and a *Remote Management Service* for smart home service providers. This middleware is used, for example, by Miele [11] to connect different household appliances to deliver automated services based on user interaction or environmental changes, like starting the vapor departure hood when the stove or oven are used or starting the washing machine when power is cheaper. ProSyst based environments are hosted locally in the user's home and can be monitored or controlled with mobile devices via web interfaces. PCEICL puts the user assistance in focus and wants to migrate the hosting environment to the cloud. ProSyst also supports E-Health scenarios [12] and is involved in several Ambient Assisted Living projects, like SOPRANO [4] or universAAL [5].

In the field of AAL, there are currently two projects also developing a platform or a middleware which realize an

 ISBN: 978-1-61208-388-9

assistance-system for elderly people: *SOPRANO* and *universAAL*.

The *SOPRANO **S**ervice **O**riented **PR**ogrammable sm**A**rt enviro**N**ments for **O**lder Europeans (SOPRANO)* project developed an open middleware for AAL solutions. The *SOPRANO Ambient Middleware (SAM)* receives user commands or sensor data, enriches them semantically and determines an adequate system response, which is then performed by the connected actors installed in the living environment. If, for example, SAM receives the information that a window is open, it analyses the remaining context information and can inform the user about the open window before he is leaving the house. The components communicate over semantic contracts and are based on a common domain ontology. This ontology is designed state-driven and every concept (device, person, location, etc.) of the ontology is represented by its actual state. The PCEICL platform, on the other hand, focuses on the user. The most important is to describe the user, since for information retrieval the user's condition is essential.

The ***UNIVER**sal open platform and reference **S**pecification for **A**mbient **A**ssisted **L**iving) (universAAL)* project [5] aims to join different approaches from lots of projects to a unique AAL solution. One of this included projects is SOPRANO [4]. The goal of universAAL is a platform, that makes it viable to develop AAL services. To meet this requirement, there will be developer tools, a store for distributing AAL services and a runtime environment to support all stakeholders. The universAAL platform is based on OSGi and ontologies are used as a common language for the components, too. But due to the OSGi-agent combination and the use of cloud technologies the PCEICL platform is more flexible and intelligent than the universAAL or the SOPRANO approach. The advantages of using agents within the OSGi framework and the usage of cloud technologies are described in Section VI.

Software agents are used in the *Emergency monitoring and prevention (EMERGE)* project [13] by the *Event-driven Activity Recognition System (EARS)* to process detected events by sensors to system reactions. Sets of detected events can be combined to assumed activities of the user. Self-StarMAS [14] uses agents to automate the configuration of devices in complex AAL scenarios. In the PCEICL approach JADE agents are used to deliver information from the user-centered ontology and to adapt the user experience and assistance by intelligent decision making.

III. Combining OSGi and Agents

To tailor the functionality and information presentation according to the user needs, we integrated the software agent platform JADE for supporting intelligent behavior with the OSGi platform, which is often basis of smart home infrastructures.

There are several related studies that deal with the combination of OSGi and JADE. They can be divided into two main approaches: a) Deploying behavioral patterns of agents as bundles in OSGi on top of the agent system or b) setting up the JADE system as an OSGi bundle.

a) Gunasekera et. al introduced VERSAG (VERsatile Self-Adaptive aGents), a general architecture for lightweight flexible multi-agent systems (MAS) [15]. Based on this architecture they present a concrete implementation for JADE and OSGi [16] (see right part of Figure 1).

In the VERSAG approach, software agents have the ability to share their *Capabilities* with other agents. *Capabilities* are OSGi bundles, that run within the agents. The kernel is managing the agents and can pass the control over to other modules. The *Itinerary Service* manages the route for the mobile agents and provides them with the needed information about the distinct location, in which the agents have to operate. The *Capabilities Repository* holds all information about the application specific *Capabilities*. The OSGi container is located in the *Capabilities Execution Service* and runs *Capabilities* that are available in the *Capabilities Repository*. In addition, the *Capabilities Execution Service* is hosting an *Adaption Service* that contains the logic for adapting the agent according to the context, and a *Context Service* that influences the aforementioned adaptation by providing context information. The configuration and implementation of VERSAG requires several significant changes to the OSGi platform, to use it within the agent.

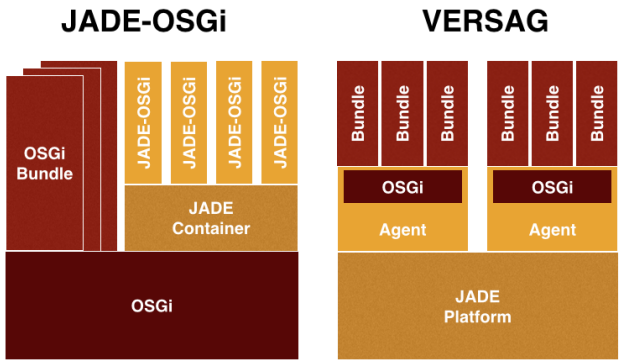


Fig. 1. Overview VERSAG Architecture

b) In Carneiro et. al [17], the JADE-system is running as an OSGi bundle and controls and manages other JADE-OSGi bundles that are independent software agents.

Jaszczyk and Król [18] follow the same approach, but divide the devices into three tiers: 1.) *Agentless Devices*, that do not have the power to provide a runtime environment with agents but have interfaces to receive FIPA-based [19] Agent Communication Language (ACL) messages [20], 2.) *Agent Devices* that are powerful enough to provide their own OSGi platform for software-agents or auxiliary OSGi bundles, and 3.) one *Main Device* that is a JADE Main Container where all other JADE Containers are registered.

Telecom Italia has developed an official JADE-OSGi bundle since JADE version 3.7, that is compatible with OSGi compliant frameworks since version 3.4 [21]. The installation and minimal configuration of JADE-OSGi is fairly easy and fast. The basic idea of JADE-OSGi is that each agent is located in a single bundle (see left part of Figure 1). The advantage of this is the possibility to use standardized OSGi actions, e.g., update agents separately during runtime by using the OSGi update functionality or deploying a *Management Agent* by using the *Configuration Admin Service*. Besides this, it is also possible to deploy other OSGi bundles in this environment

 ISBN: 978-1-61208-388-9

and agents can access the services of these bundles. This leads to the possibility to keep the agents simple by reusing functionality that is provided by other bundles. The dynamic and modular architecture of OSGi is extended by JADE-OSGi and introduces additional intelligence by adding flexible agents to the framework.

The approach b) setting up the JADE system as an OSGi bundle, is also the approach taken in the PCEICL platform.

IV. Platform Architecture of PCEICL

The PCEICL platform combines the modular service platform OSGi (Equinox [22]) and intelligent software agents using JADE-OSGi [21]. The platform is built up on an infrastructure, that is provided by the Cloud Management System (see Section V). First an architecture overview is given, second an architecture layer model is described and finally it is shown, how the PCEICL ontology is integrated into the platform.

A. Architecture Overview

PCEICL architecture is based on the OSGi platform with several OSGi bundles for common services, like *Sensor Bundles*, *Smart Home Control Bundles*, an *Address Book Bundle* or a *Web-Interface Bundle* (see Figure 2).

JADE-OSGi is hosting the software-agents system but can also register and communicate with agents, that are OSGi bundles themselves. Only the agents have access to the PCEICL-Ontology [8] and pass the aggregated data to the service bundles. Authentication and access control is managed by the *Authentication* and *Security Module* that is also securing the OSGi framework. For example, a *Message Access Control Module* controls the messages flow between the various OSGi bundles in detail. Special OSGi bundles provide services for installing new bundles and functionality or for updating existing ones through a central *Bundle Repository* as described in Section V.

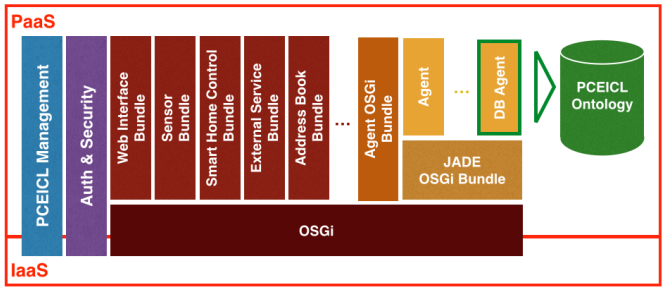


Fig. 2. Overview of PCEICL Architecture

B. Architecture Layer Model

The architecture of the PCEICL platform can be divided into three different layers (see Figure 3):

- *Agent Layer*
  All agent bundles are located in the *Agent Layer*. Communication between agents within this layer is performed by Agent Communication Language (ACL) [20].
  There are *Application Agents*, like the *Reminder Agent*, which can be developed by external developers and provide several services for the user. Other agents are *System* and *Smart Agents*, which are part of the PCEICL platform. *Smart Agents* are for example domain expert agents, which have access to the domain ontology data and are able to execute ontology reasoning. *System Agents* can be used by other agents for system matters, like access control or sensor data acquisition.
- *OSGi Bundle Layer*
  In this layer, all OSGi bundles are placed, that provide services. These bundles are usually provided by smart home service providers. They integrate sensors, like temperature sensor, window status, etc.
- *External Service Layer*
  This layer provides services, that can be accessed from outside of the OSGi platform, e.g., a weather web service.

The communication between the *Agent* and the *OSGi Bundle Layer* is implemented by the OSGi framework itself.

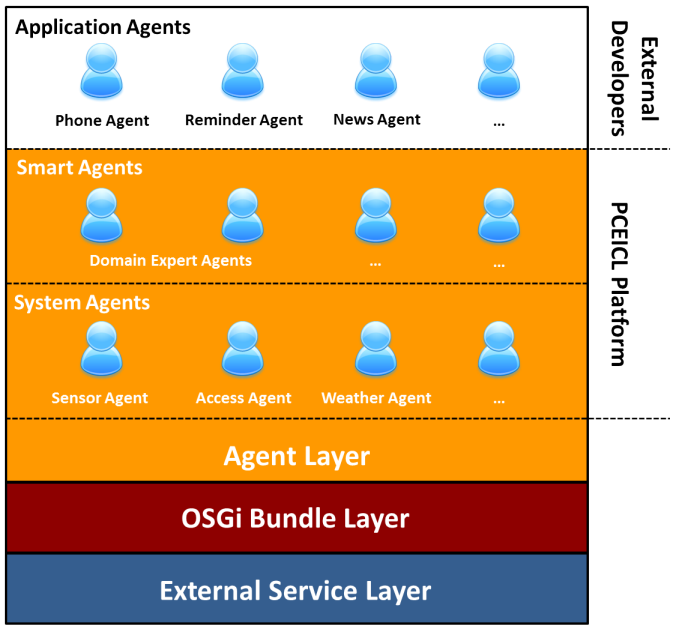


Fig. 3. Layers of the PCEICL Platform

C. User-Centered PCEICL Ontology Integration

Based on the PCEICL-Ontology [8], the user context can be semantically interpreted and can be used directly by the ACL for exchanging messages. This is used to adapt all services of the PCEICL platform to the needs of the user. The PCEICL ontology is, unlike in other approaches, user-centered. This means, that it models the user and his properties like personal information, interests, preferences, health condition but also the user's environment (social contacts or information about sensors, devices, weather, etc.). Additionally, the ontology is easy to expand and offers the possibility to have a historical view over the changing user data. To achieve access control of

 ISBN: 978-1-61208-388-9

the PCEICL-Ontology only the *PCEICL-Ontology DB Agent* is allowed to access the stored information about the user.

V. Platform in the Cloud

PCEICL is a specialized platform, which delivers specific core functionalities in the field of Ambient Assisted Living as a service. It supports common home automation scenarios but also adds another layer of interaction by services and user interfaces to assist the user in his daily life.

Each environment is separated from other environments for privacy reasons and is managed by a *PCEICL PaaS Management System*. The *PCEICL PaaS Management System* has the following functions:

- *Multi-Tenant:* Multiple customers can use the PCEICL PaaS.
- *Customizable:* The PCEICL PaaS is customizable. For instance, if the system is used in a facility for assisted living, special devices need to be preconfigured during the installation of the PaaS. This is done by installing the needed bundles and services for these devices without the participation of the user. After this, the user can customize additional functionalities or install the services that assist him in his daily life. The user can proceed on his own or is helped by trained staff during the initial setup of the PCEICL platform.
- *Scalable:* The *PCEICL PaaS Management System* can also monitor the workload of the PCEICL PaaS instance and request additional resources at the *Cloud Management System* locally or remotely at the Public Cloud.
- *Granular Security Control:* The PCEICL PaaS offers a flexible access control system that allows detailed control over the user's data access and services access.

The private clouds share centralized services, hosted in the Public Cloud, like the aforementioned *Cloud Management System* or *Community Event Services*.

A. OSGi Bundle Repository

An *OSGi Bundle Repository* based on OBR [23] provides new services or updates existing services and agents (see Figure 4). In this example the Private Cloud is reporting a misbehavior or error inside a service/agent through a *Reporting Module*. This malfunction can be solved by an update of the affected services. After the update is released to the central *Bundle Repository*, the *PCEICL PaaS Management System* of each Private Cloud is notified that there is a new version of the service ready for deploying. The *PCEICL PaaS Management System* then triggers the update on the PCEICL platforms. The platforms evaluate if the update is part of a service, that is running in the platform, and if the bundle is still in use by an active service. If it is in the active bundle set but currently not in use, the PCEICL platform pulls the updated bundle from the Bundle Repository and deploys it. Otherwise the new bundle is installed alongside the existing and currently running bundle. When the deployment of the new bundle is finished, the old bundle is stopped and deleted and new requests will be processed by the new bundle. This update process is implemented to keep the availability as high as possible.

The central *Bundle Repository* is also accessed by the installation bundle for new services. The first time this is used, is during deployment of PCEICL by the *PCEICL PaaS Management System* and during setup of the individual PaaS instance through the primary user. Through the central *Bundle Repository*, all PaaS instances are on the same software version. This contributes to future development and maintainability, because the currently deployed bundles are the same across all PCEICL platforms and can be used as a basis for new bundles and services.

B. Reporting Module

Besides the already mentioned service bundle update reporting, the *Reporting Module* has an anonymized information interface. This interface is used to allow the improvement of software agent reasoning in the PCEICL platforms. For example, the suggestion of new services based on the installation setup of other PCEICL platforms can be realized. But, further ideas, like the collection of anonymized data about the user behavior, can help to improve the PCEICL services.

VI. Particularities of the PCEICL Platform

Based on the presented technology the PCEICL architecture is context aware and supports intelligent behavior. Especially the platform flexibility is worth mentioning and will be pointed out next.

- *Flexibility through Private Cloud:*
  New PCEICL service instances can be created or deleted on demand. This is relevant in settings of large facilities for assisted living that aim to keep their inhabitants living a self-determined life as long as possible. The data of the user (e.g., profile, configurations, etc.) are hosted locally.
- *Flexibility through PaaS:*
  Based on the load of the platform, it can react by requesting more resources through the PCEICL Management System and the attached monitoring. The PaaS is able to grow with the user's demand on assistance or services. Detailed access control is part of the platform.
- *Flexibility through Service Adaptability:*
  Each PCEICL platform can be set-up with different services and use-cases in mind. Home-automation scenarios based on OSGi are possible and can be extended with assistance systems to support the user in his daily life. The user has the tools to customize the user-experience by adding services or trying out new application offers. The system itself has the flexibility to support the user by responding to new requirements based on the behavior or health state of the user.
- *Flexibility through Functional Adaptability by Context Awareness:*
  The service functionality varies according to the context awareness [24] of the system. The system and its services have the ability to communicate with sensors in the living environment or with external services,

 ISBN: 978-1-61208-388-9

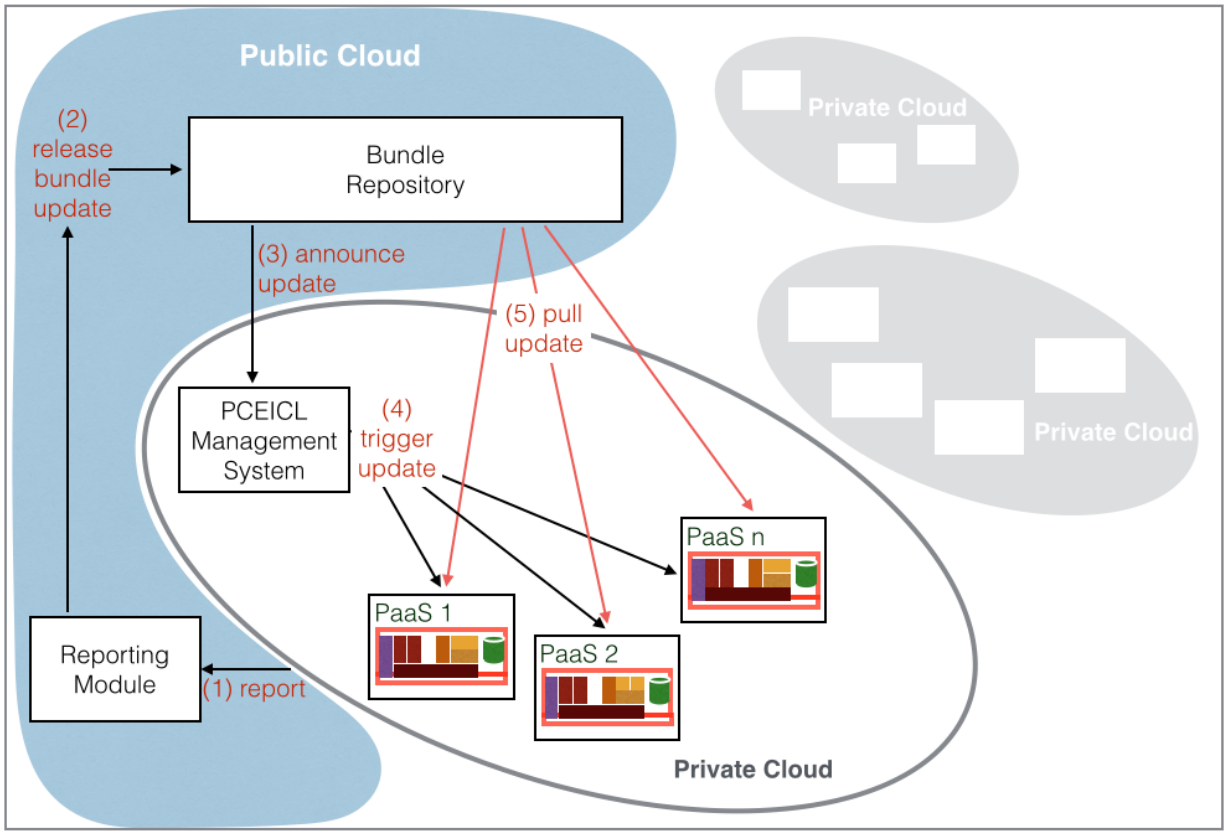


Fig. 4. Private clouds share centralized services in the Public Cloud. Example: Private Cloud updates bundles from the Public Cloud Bundle Repository based on the Reporting Module.

like, e.g., weather forecasts or calendars of events, to react with special features or services. It also has the ability to take the user's condition into account and adapt the services or to simply alert trained personnel for help.

PCEICL platform therefore uses the technology of agent systems to make the right decision at the right time. These lightweight and intelligent software agents can be updated and refined based on the anonymized data that can be retrieved by the Reporting Module.

VII. Evaluation

For a first evaluation of the architectural approach, we follow the Software Architecture Analysis Method (SAAM) [25] using the SAAM step: *Perform Scenario Evaluation*. The developed four scenarios try to evaluate the main features of the presented PCEICL architecture.

A. Scenario 1: Adding New Functionalities by the User

The direct scenario of adding new functionalities to a PCEICL instance by the user involves the private PaaS of the user and the *Bundle Repository*. The user has access to a central market place AAL OSGi bundle repository, that will list different applications, that can be added to the platform. The user chooses the new functionality. The *PCEICL PaaS Management System* requests for the new bundle at the *Bundle Repository*. This bundle requires a set of bundles. The platform evaluates, which required bundles are already running on it and which have to be added for the new functionality. If all required bundles and the bundle that contains the desired functionality, are deployed on the user's PCEICL platform, the functionality is started and ready for further configuration, e.g., specifying login details or changing color schemes based on the user's needs.

Involved PCEICL platform modules: *Bundle Repository*, *PCEICL PaaS Management System*

B. Scenario 2: Adding Resources to a PCEICL Instance

Adding resources to a PCEICL PaaS, when predicting performance shortcomings, is another direct scenario that is fully supported by the presented architecture. The *PCEICL PaaS Management System* continually monitors the platforms inside the private cloud. If an increase in demand for resources is measured or estimated, the *PCEICL PaaS Management System* can request additional resources, e.g., memory or CPU, at the *Cloud Management System*. The Cloud Management System is responsible for the IaaS layer and can then assign the resources to the platform.

Involved PCEICL platform modules: *PCEICL PaaS Management System*, *Cloud Management System*

 ISBN: 978-1-61208-388-9

C. Scenario 3: Updating a Bundle

As described in Section V and shown in Figure 4, the updating process of a bundle is supported by the architecture, which could imply a direct scenario. However, an updated bundle is changing functionality of the platform itself and is able to interact with other modules. This characterizes an indirect scenario in SAAM.

If a developer wants to update a bundle, he can use an instance of the PCEICL platform to test the functionality against it before introducing it in the Bundle Repository. Changes in bundles, which are only loosely coupled with other services, are simpler to update than closely coupled services. Changes in core services lead to a greater impact on other bundles and have to be treated very carefully. The difficulty of each change therefore has to be evaluated case by case. The modularity of the system however supports the process of updating bundles and the architecture can provide the aforementioned testing platform.

Involved PCEICL platform modules: *Bundle Repository*, *PCEICL PaaS Management System*

D. Scenario 4: Changing System Behavior Based on Context Changes of the User

This direct scenario describes the context awareness of the system using a simple *Ride Offering Service*. The user has added an event to his calendar that he wants to attend. After he had broken his leg in an accident, his health state in the PCEICL system (*PCEICL Ontology DB*) is updated by his doctor through the *PCEICL Ontology DB Agent*. The system reacts to this new state by offering a request for a lift to the event. This is also the adapted behavior for all new events that he would like to attend. If the health state changes again and the user is mobile again, the system can take back the changes and respond as before the accident.

Involved PCEICL platform modules: *PCEICL Ontology DB*, *PCEICL Ontology DB Agent*, (scenario specific: *Ride Offering Service*)

VIII. Conclusion and Future Work

In this paper, we presented an architecture approach for a context aware, intelligent and flexible PaaS for the use in Ambient Assisted Living.

The PCEICL platform is based on OSGi and extended by the JADE agent system. The agent system has access to the *PCEICL-Ontology* and reacts to changes by adapting the services, that are run on the platform. PCEICL is designed as a Platform as a Service, has a *PCEICL Management System* and runs in a private cloud for security and privacy reasons. The *PCEICL PaaS Management System* has the ability to manage the platform, preconfigure, start and stop services on demand, does load balancing and monitors the resources. Several common services, that are used by the PCEICL platform, run in the Public Cloud and can be accessed from all other platforms. The user has the ability to configure the platform to his needs and install new services from a central *Bundle Repository*. The Bundle Repository contains new software bundles, updates for existing bundles, special bundles with agents included, etc.

The paper discussed the improvement of flexibility by combining OSGi and cloud mechanisms. At all layers of the PCEICL architecture, this approach can improve the platform for the field of AAL.

Most of the data, that is stored in PCEICL Ontology DB, is only accessible by the user. Future work would be to refine the access of the data by the agents, based on a Role Based Access Control (RBAC), so that only services, that are trusted, get access to the data they need to deliver the service. These roles should be transparent to the users and developers of new services. One promising approach could be the integration of JADE-S [26] to dynamically assign permissions to agents.

The data privacy is also a matter for discussion, when collecting data by the *Reporting Module*. How can data be anonymous but still being significant across several private cloud infrastructures? This has to be considered, when trying to update services and agents to improve the experience of the platform.

Acknowledgment

The project ZAFH-AAL (“Zentrum für Angewandte Forschung an Hochschulen für Ambient Assisted Living”) is funded by the Ministry of Science, Research and the Arts of Baden-Württemberg, Germany. The funding program for the universities of applied science is called: Zukunftsoffensive IV “Innovation und Exzellenz” (ZO IV). The PCEICL project is a sub-project of the project ZAFH-AAL [1].

References

[1] “ZAFH-AAL - Zentrum für angewandte Forschung an Hochschulen für Ambient Assisted Living (Collaborative Center for Applied Research on Ambient Assisted Living),” http://www.zafh-aal.de, [retrieved: 2014.07.18].

[2] “PCEICL a Person Centered Environment for Information, Communication and Learning,” http://www.wolke.hs-furtwangen.de/currentprojects/pceicl, [retrieved: 2014.07.18].

[3] OSGi Alliance, “Smart home market,” http://www.osgi.org/Markets/SmartHome, [retrieved: 2014.08.02].

[4] M. Klein, A. Schmidt, and R. Lauer, “Ontology-centred design of an ambient middleware for assisted living: The case of soprano,” http://publications.andreas.schmidt.name/klein_schmidt_lauer_AIM-CU_KI07.pdf, [retrieved: 2014.07.11] 2007.

[5] R. Ram et al., “universaal: Provisioning platform for aal services,” in Ambient Intelligence - Software and Applications, ser. Advances in Intelligent Systems and Computing, A. Berlo, K. Hallenborg, J. M. C. Rodríguez, D. I. Tapia, and P. Novais, Eds. Springer International Publishing, 2013, vol. 219, pp. 105–112.

[6] Fraunhofer Institute for Open Communication Systems - FOKUS, “AAL-Kompetenz - The information portal for developers of intelligent assistance systems,” http://www.aal-kompetenz.de/, [retrieved: 2014.07.08].

[7] AALIANCE, Ambient Assisted Living Roadmap - AALIANCE Project - Deliverable 2.7, March 2010, ch. Enabling Technologies, pp. 95–96.

[8] C. Fredrich, H. Kuijs, and C. Reich, “An ontology for user profile modeling in the field of ambient assisted living,” in SERVICE COMPUTATION 2014, The Sixth International Conferences on Advanced Service Computing, A. Koschel and A. Zimmermann, Eds., vol. 5. IARIA, 2014, pp. 24–31.

[9] F. Bellifemine et al., “Java agent development framework,” http://jade.tilab.com/, [retrieved: 2014.07.12].

[10] ProSyst, “Smart home / smart energy,” http://www.prosyst.com/what-we-do/smart-home-smart-energy/products/, [retrieved: 2014.07.15].

 ISBN: 978-1-61208-388-9

[11] Miele & Cie. KG, "Miele@home," http://www.miele.de/haushalt/hausgeraetevernetzung-1912.htm, [retrieved: 2014.08.03].

[12] M. Petzold, K. Kersten, and V. Arnaudov, "Osgi-based e-health / assisted living," ProSyst, http://http://www.prosyst.com/fileadmin/ProSyst_Uploads/pdf_dateien/ProSyst_M2M_Healthcare_Whitepaper.pdf, Whitepaper, September 2013.

[13] H. Storf et al., "An event-driven approach to activity recognition in ambient assisted living," in AmI 2009, M. Tscheligi et al., Ed. Springer Verlag Heidelberg, 2009, pp. 123–132.

[14] I. Ayala, M. Amor, and L. Fuentes, "Self-starMAS: A multi-agent system for the self-management of AAL applications," in Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IEEE. IEEE International, 2012, pp. 901–906.

[15] K. Gunasekera, A. Zaslavsky, S. Krishnaswamy, and S. W. Loke, "VERSAG: Context-aware adaptive mobile agents for the semantic web," in COMPSAC '08. 32nd Annual. IEEE International, July 2008, pp. 521–552.

[16] K. Gunasekera, A. Zaslavsky, S. Krishnaswamy, and S. W. Loke, "Building ubiquitous computing applications using the VERSAG adaptive agent framework," Journal of Systems and Software, vol. 86, no. 2, pp. 501 – 519, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121212002695

[17] D. Carneiro, P. Novais, R. Costa, and J. Neves, "Developing intelligent environments with OSGi and JADE," in Artificial Intelligence in Theory and Practice III, ser. IFIP Advances in Information and Communication Technology, M. Bramer, Ed. Springer Berlin Heidelberg, 2010, vol. 331, pp. 174–183.

[18] P. Jaszczyk and D. Król, "Updatable multi-agent osgi architecture for smart home system," in Agent and Multi-Agent Systems: Technologies and Applications, ser. Lecture Notes in Computer Science, P. Jedrzejowicz, N. Nguyen, R. Howlet, and L. Jain, Eds. Springer Berlin Heidelberg, 2010, vol. 6071, pp. 370–379.

[19] "The foundation of intelligent physical agents," http://www.fipa.org/, [retrieved: 2014.07.08].

[20] "Agent communication language specifications," http://www.fipa.org/repository/aclspecs.html, [retrieved: 2014.07.12].

[21] E. Quarantotto and G. Caire, "JADE OSGi GUIDE," http://jade.tilab.com/doc/tutorials/JadeOsgiGuide.pdf, April 2010.

[22] The Eclipse Foundation, "equinox OSGi," http://www.eclipse.org/equinox/, [retrieved: 2014.07.14].

[23] W. J. Gédéon, OSGi and Apache Felix 3.0, 1st ed. Packt Publishing, November 2010, no. 978-1-84951-138-4, ch. Using the OSGi Bundle Repository.

[24] A. K. Dey and G. D. Abowd, "Towards a better understanding of context and context-awareness," in Computer Human Intraction 2000 Workshop on the What, Who, Where, When, Why and How of Context-Awareness, 2000, pp. 304–307.

[25] R. Kazman, L. Bass, G. Abowd, and M. Webb, "SAAM: A Method for Analyzing the Properties of Software Architectures," Software Engineering Institute, Carnegie Mellon University, White Paper, May 2007.

[26] JADE Board, "JADE Security Guide," http://jade.tilab.com/doc/tutorials/JADE_Security.pdf, February 2005.

 ISBN: 978-1-61208-388-9

Cloud Management Platform Selection: A Case Study in a University Setting

Francisco Cocozza, Gustavo López, Gabriela Marín, Ricardo Villalón and Francisco Arroyo

Centro de Investigaciones en Tecnologías de la Información y Comunicación (CITIC)
Universidad de Costa Rica
San José, Costa Rica
Email: {francisco.cocozzagarro; gustavo.lopez_h; gabriela.marin; ricardo.villalon; rafael.arroyo}@ucr.ac.cr

***Abstract*—Cloud Computing has gained importance in recent years. There are many implementations' analyses and evaluations of Cloud Management Platforms (CMPs) in the literature. Moreover, the context and characteristics differ drastically between implementations, depending on the user requirements and usage context. This paper presents a case study of the process we followed to select a Cloud Computing management platform to be deployed in a University. Administrative and academic requirements were gathered and studied to define the most appropriate platforms. We present an overview of available CMPs. Moreover, we show a set of comparison criteria that could be used to determine which CMP adapts best to the cloud deployment scenario.**

Keywords–Cloud Computing; VCL; OpenStack; Cloud Management Platform.

I. INTRODUCTION

Requirements for computational services in the industry and academia have grown vastly in the past decades. Cloud Computing is one of the efforts conducted by the information technologies and computational scientists to keep up with the demand. The National Institute of Standard and Technology (NIST) of the U.S. Department of Commerce defines Cloud Computing as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction [1]. The essential characteristics of Cloud Computing are: (1) on-demand self-service, (2) broad network access, (3) resource pooling, (4) elasticity, and (5) measured quality of service [2].

There are many possible settings for a cloud, e.g., universities, commerce or multinational companies. Even though nowadays there are several public clouds offering services, we consider important to describe the process of developing a cloud depending on the context in which it will be deployed.

Our research is motivated by the fast growth of Costa Rican industry in areas related to technology and services. Since 1998 more than 100 companies focused on IT related services, established operations in Costa Rica, including companies like IBM, Sykes, Infosys, VMWare, Hewlett-Packard, Intel and others. Moreover, 5.8% of national GDP consists of IT & IT Services. Without doubt, Cloud Computing is currently undergoing a huge hype and many companies see it as the future of IT [3]. This situation creates a constant demand of trained and qualified professionals. Being the major University in the country, University of Costa Rica (UCR) needs to deal with the training of these professionals. In order to train professionals in Cloud Computing and related topics, the University required to implement its own cloud services, and create know-how on the trending topics. The main effort conducted by UCR was deploying a cloud and create coursework for the topic.

These paper aims to provide an experience report and a set of tools that could be helpful when deploying a cloud. This research project was conceptualized after a training in which the benefits of Cloud Computing for universities were presented. The presenter explained how Cloud Computing through the implementation of Virtual Computing Lab (VCL) improved IT services at North Carolina State University. Several research papers were published by VCL partners. They demonstrated how VCL reduced costs and help delivering computational resources to over 30,000 students and faculty members [4][5]. We decided to reproduce their effort at UCR.

A cloud is a complex entity wich consists of several components such as hypervisors, authentication mechanisms, file system backends, certificate authorities, data base engines, and others. An implementation of such elaborated infrastructure involves an extensive number of decisions, which cannot be covered and described on a single paper. Given that, this work is focused on the process we followed to select the software platform for managing the UCR's cloud.

In the process of creating our cloud, we found several CMPs comparisons in the literature. However, these comparisons vary greatly in terms of granularity and the concepts addressed. Therefore, we proposed a set of criteria to consider when implementing a cloud. Our criteria are based on literature review and expertise obtained while implementing UCR's cloud.

The comparison of CMPs is a complex labor, identify the characteristics of software depends on the completeness of its documentation and case studies reported. Therefore, from all the considered CMPs we applied our comparison criteria only in four preselected ones: CloudStack, Eucalyptus, VCL and OpenStack.

The rest of the paper is structured as follows. Section two shows the related work, focused on an overview of available CMPs and comparisons reported in literature. Section three gives the context in which UCR's cloud was developed. Section four shows the process we followed to select a CMP and the key features that guided our selection. Section five shows the selection process results and discussion. Finally, Section six shows some conclusions and future work.

 ISBN: 978-1-61208-388-9

II. Related Work

According to a recent Merrill Lynch research note [6], Cloud Computing is expected to be a $160-billion addressable market opportunity. Cloud Computing is also a prominent technology trend [7]. Given this large market, many companies with a desire for profit and other nonprofit organizations presented a variety of CMPs in the past years.

A CMP is a set of software for managing cloud environments [8]. A CMP includes self-service interfaces, provisioning system images, usage measure and workload optimization. CMP's main goal is to allow enhanced resource management and monitoring of the cloud resources.

Several CMPs are available nowadays: Abiquo [9], CloudStack [10], Eucalyptus [11], Nimbus [12], openQRM [13], Openstack [14], Open Nebula [15], Apache Virtual Computing Lab (VCL) [16], HP's CloudSystem Matrix [17], among others. In this Section, we present a brief introduction to these CMPs and their main deployment scenarios.

Abiquo [9] is a hybrid cloud management system for small and medium-sized business. Abiquo supports multiple hypervisors and is focused on enabling organizations to leverage existing virtualization technologies and public clouds. Abiquo also allows central management of resources via Graphical User Interface (GUI).

Apache CloudStack [10] is an open source software designed to deploy and manage large networks of virtual machines as Infrastructure-as-a-Service (IaaS) Cloud Computing platform. CloudStack is used by service providers running cloud services, product vendors and organizations who have used the software to deploy private clouds.

Eucalyptus is a Linux based open source software. Eucalyptus is the acronym for Elastic Utility Computing Architecture Linking Your Programs to Useful Systems. It was developed for creating private and hybrid clouds. The software is suited for enterprise clouds that supports the industry-standard. [18][11].

Nimbus is a set of open source software Cloud Computing components written in Java and Python focused on providing IaaS capabilities to the scientific community [12]. Nimbus is designed to turn clusters into an Infrastructure as a Service cloud.

OpenQRM [13] is a free and open-source Cloud Computing management platform for managing heterogeneous data center infrastructures. The openQRM platform manages a data center's infrastructure to build private, public and hybrid IaaS clouds. OpenQRM is designed for companies of various kinds.

OpenStack is a free and open source Cloud Computing software platform. OpenStack software controls large pools of compute, storage, and networking resources. OpenStack is designed to be deployed in many settings. OpenStack's use is largely reported in various research papers [19][20][21]. Moreover, 35 case studies are reported by OpenStack in the use of the software [14].

OpenNebula [15] is an open source project aimed at building the industry standard open- source Cloud Computing tool to manage the complexity and heterogeneity of large and distributed infrastructures.

Apache VCL [16] is an open-source solution for the remote access over the Internet to dynamically provision and reserve computational resources for diverse applications, acting as Software as a Service (SaaS) solution. VCL was conceived as a tool for educational use and was first deployed at East Carolina University, USA.

HP CloudSystem Matrix [17] is a cloud infrastructure from Hewlett-Packard that combines storage, servers, networking and software for organizations to build complete private, public and hybrid Cloud Computing environments.

Knowing the quantity of CMPs available several researches have been conducted to compare different CMPs. However, these comparisons are too different from one another to be used together when choosing a cloud management platform. For instance Cordeiro et al. [22] compare Xen cloud platform, Eucalyptus and OpenNebula. The comparison is based on the platform's architecture, their networking management, virtual machine placement and inter-host communication. On the other hand, Voras et al. [23][24] present a set of comparison criteria based on storage, virtualization, management, network, security and support. The authors introduce several cloud management platforms and other cloud related software such as Open Nebula, Eucaliptus, Ubuntu Enterprise Cloud, OpenQRM, Abiquo, Red Hat Cloud Foundations, Edition One, OpenStack, Nimbus, mOSAIC. However, no evaluation is reported.

Wind [25] presented another comparison between Eucalyptus, OpenNebula, Abicloud and Nimbus based on these criteria: architecture, programming language, supported cloud types and hypervisors, user interface, licensing, robustness, interoperability, security and compatibility. This research presents an interesting comparison criterion, however, only four CMPs were evaluated and the more used platforms were not included. The criterion used in our research includes several of the criteria presented by Wind, but applied on other CMPs. Cerbelaud, Garg and Huylebroeck [26] compared Enomaly Elastic Computing Platform (ECP) Eucalyptus, OpenNebula and oVirt based on the following criteria: VM creation tool and repository, image storage, uploading, saving and choosing host. Steinmetz, Perrault, Nordeen, Wilson and Wang compared OpenStack and Eucalyptus; however, their measurement was solely based on performance.

Other authors have compared cloud vendors and providers. For instance, Khan, Noraziah, Herawan, and Mat Deris [27] compared Amazon EC2, Microsoft Azure, Google App Engine, Sun Grid and GRIDS Lab Aneka based on: service type, user access, virtualization capabilities, and programming framework. A similar comparison was made by Li. Yang, Kandula and Zhang [28] they compared Amazon AWS, Microsoft Azure, Google AppEngine and Rackspace CloudServer. However, the authors used a totally different set of criteria, making impossible any comparison between the two. The last choose elasticity, storage, internal communication, networking and cost as their criteria.

This section showed the variety of CMPs and cloud vendors in the market. Moreover, we introduced most of the currently available CMPs. On the other hand, we presented some comparisons available in the literature and proved that the compared CMPs and criteria used are very difficult if not impossible to compare. The main differences found were:

- Heterogeneity: we discovered different terms referencing the same concept. This can be associated with

 ISBN: 978-1-61208-388-9

the lack of standardization on cloud platform features names.

- Granularity: some comparisons provided detailed criteria about a specific feature, while other comparisons evaluated high level characteristics of the cloud platform.
- Completeness: the criteria did not meet our expectations on the level of detail and covered features that we wanted to evaluate.

Given the lack of homogeneity and completeness, and the difference in granularity, we decided to use our own set of criteria. Our definition is based on previous comparisons and our contextual requirements.

III. CONTEXT

This section details the context and background in which the UCR's cloud was developed and presents an overview of processing power and capacity required by the cloud services to be deployed.

A. Demographics and physical contexts

UCR is the largest and most important Costa Rican university. Established in 1940, it has three main focuses: research, teaching and being socially responsible. It is a public university with a budget of $454 million USD for 2014 [29]. UCR's academic offer includes 244 undergraduate programs and 243 graduate programs (including masters and doctoral programs).

With over 40,000 active students, 4,292 faculty members and 3,520 administratives, UCR has large computational requirements. UCR is spread all over Costa Rica. The main campus is in the capital of the country, also offers seven secondary campuses and four local branches (smaller facilities). The IT Department in charge of supplying these requirements was starting to exceed their capacity.

UCR's physical and demographical complexities lead us to consider all these factors when creating a computational solution that meets academic and administrative requirements.

B. University IT Services

The services provided to the academic and administrative population of the University are in charge of the IT Department. It offers a variety of services ranging from networking and infrastructure maintenance to website hosting.

The IT Department implemented, some years ago, virtualization hosting services to fulfill the diverse university's requirements. As the number of requests increased, the services performance and capacity became a problem. The main problem was that the hardware was not able to keep up with the demand. With 140 virtual machines distributed among eight virtualization servers, the platform had reached its limit by the end of 2012. In 2013, the IT Department deployed a strategic plan for continuous improvement in which they decided to update their hardware.

The main services provided by the IT Department are: web hosting, virtualization, mail services, DNS services, network infrastructure maintenance, database management, computational equipment provisioning. Nowadays, these services are manually requested and implemented; therefore, any automation would be beneficial.

One of the main goals of the IT Department is to decentralize the administration of the resources they provide. This goal can be achieved by offering self-provisioning services (one of the main features of a cloud). NIST defines self-provisioning or self-service as: the ability of a user to unilaterally provision computing capabilities, without human interaction of the service provider [1]. Self-provisioning is a particularly important aspect because it enhances the perception of service's efficiency and makes the management and use of the resources easier for both, the service provider and the end user.

Any change in IT services causes the raise of disagreements. We faced this reality when implementing UCR's cloud platform. Based on the research presented by Kuo [30], we established the main challenges and opportunities for this cloud implementation (Table I).

Organizational inertia relates to the resistance to change by not only the end users, but also the people in charge of maintaining and supporting the services. The lack of experience related to Cloud Computing was also a challenge. Even though most of the people implementing the cloud are experienced professionals in virtualization, network and infrastructure areas, they did not have proper training in Cloud Computing at the beginning of this project.

Costa Rican jurisdiction on digital data is still very ambiguous. This causes security and data jurisdictional concerns needed to be addressed in order to use international cloud services. Moreover, University of Costa Rica's infrastructure may cause a bottleneck that could be a challenge when deploying cloud services.

All the challenges mentioned above were considered; however, we had an opportunity windows that we were encouraged to use. The main opportunities were: financial support of high authorities at UCR and a vision for change in the administration. Many offices inside UCR requested IT infrastructure improvements, the creation of a cloud could take advantage of the existing infrastructure and share resources in order to satisfy most of these needs. Moreover, at country level, this project represents a great opportunity. Currently the IT services industry in Costa Rica is a sophisticated and continuously growing area.

According to the Costa Rican Investment Promotion Agency (CINDE) [31], companies are constantly demanding trained and qualified professionals in areas like Cloud Computing, Big Data, Virtualization, and others. These market requirements forced UCR to improve its educational offer in Cloud Computing and related technologies. Creating a cloud could provide the expertise and new technological infrastructure required for this improvement.

TABLE I. CHALLENGES AND OPPORTUNITIES CONSIDERED IN THIS CLOUD IMPLEMENTATION

Challenges	Opportunities
- Organizational inertia. - Lack of expertise in Cloud Computing. - Security and data jurisdiction issues. - Infrastructure bottlenecks.	- Financial support for hardware acquisition. - Visionary and change-aware administration. - IT Infrastructure needs. - Growing IT services industry requiring trained and qualified professionals.

 ISBN: 978-1-61208-388-9

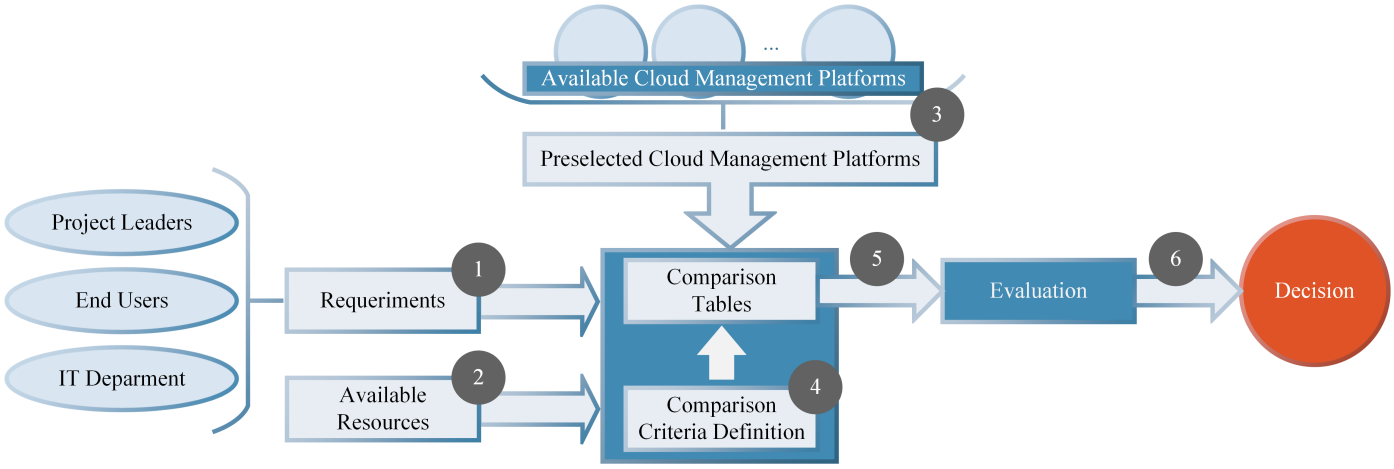


Figure 1. Process followed to select a Cloud Management Platform.

IV. Cloud Management Platform Selection Process and Implementation

This section presents the process followed to select a cloud platform used to meet UCR's computational requirements. Several cloud platforms exist. We wanted to select the most appropriate one to UCR's context, taking into account important variables given by the context, institutional regulations, legal issues and user requirements.

Figure 1 shows the components and steps involved on the process to select a cloud platform. To summarize, the process consisted of the following steps:

1) Gather the requirements and expectations for the cloud platform from different perspectives; i.e., the stakeholders.
2) Determine the resources (hardware and software licenses) already available to support the cloud platform, and determine their technical specifications.
3) Identify available cloud platform solutions and preselect a subgroup based on basic filtering, based on the most relevant criteria.
4) Define our own set of comparison criteria, based on other comparisons and information gathered in the previous steps.
5) Compare the cloud platforms based on our criteria.
6) Select the cloud platform that best fits our requirements.

The following subsections provide an in-depth description of the implementation of each one of the six steps mentioned above.

A. Requirements (step 1)

The CMP implementation is one of the projects of the IT Department. For all purposes, the IT Department plays the role of Service Provider (SP). One of their main goals is to provide high quality services that can meet the expectations and requirements of their users. As a SP, the IT Department has to ensure that the platform is flexible in its service delivery while keeping the users isolated from the underlying infrastructure [27].

Profiles for users in a university setting are highly diverse. Therefore, we determined that the requirements for this project should be gathered from different perspectives. As shown in Figure 1, the "requirement component" includes the feedback of three different groups: (1) IT Department, (2) Cloud Computing project leaders and (3) end users.

For collecting the requirements from the IT Department, different approaches were used. To understand the internal processes of the IT Department we established periodical technical meetings. After we had a clear vision of the internal functioning of the IT Department, we met with specialized technicians that could contribute in the definition of the cloud. The topics discussed in the meetings were widely varied, ranging from network topologies to virtualization technologies. The main goal of the meetings was to shape and refine the required cloud services.

The Cloud Computing project managers and researchers involved in this project were the ones considered to establish the requirements for academic purposes. The team in charge of the project includes seven computer scientists (4 Ph.D., 2 M.Sc, and one B.Sc.).

For the end users perspective, a different approach was used. In this case, a survey was applied to the local IT administrators of every department and faculty to gather their opinion and expectations on the new services. With the input provided on this survey, we identified some of the most requested services for the cloud. Figure 2 shows the main services requested and the percentage of the end users that wanted each service. The perspectives mentioned were consolidated to establish the following requirements:

The general requirements are:

- Licensing: the cloud platform must have an Open Source licensing model.
- CMP Database Management System: the selected DBMS must be one of the following: MySQL, MariaDB and/or PostgreSQL, which are Open Source DBMS.

The academic requirements are:

- Lab deployment support: the cloud must offer the ability to deploy virtual computer laboratories [32] for academic courses.

 ISBN: 978-1-61208-388-9

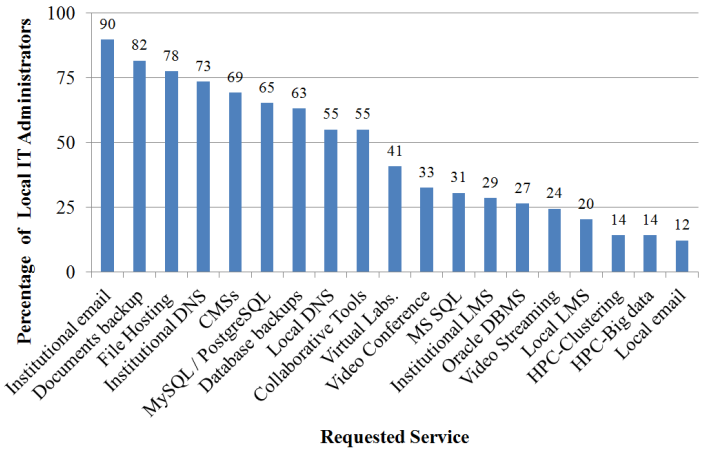


Figure 2. Cloud services requested by users (local IT Administrators)

- Bare-metal provisioning: the cloud operative system should have built-in support for automated bare-metal provisioning. Bare-metal provisioning is the process of installing an Operating System (OS) or Type 1 hypervisor directly on a computer's hardware [33].
- Scheduling of resources reservation: it covers a typical academic scenario, where the same type and number of virtual machines are needed on a specific time and on a regular basis.

Finally, the administrative requirements are:

- High availability for critical services.
- Infrastructure compatibility: the cloud platform must be aligned with the current hardware and software. For example, the currently licensed hypervisors must be supported by the cloud platform. Also, integration with specific appliances is preferred.
- Service model: the cloud platform must provide the most basic service model, Infrastructure as a Service (IaaS). This was defined by the IT Department and us (the researchers) as the initial stage of the cloud service. Based on IaaS, other service models like Software as a Service (SaaS) or Platform as a Service (PaaS) could be offered in the future.
- Platform management: the selected platform must provide features that facilitate its use and administration.
- API completeness: the services and functionality provided by the CMP's API must be extensive and exposed using standards, e.g., REST designed [34].

B. Available Resources (step 2)

The second step in the procedure is to determine the available hardware. In a cloud, the hardware plays a key role on the success and reliability of the provided services.

The IT Department at UCR assigned its newest hardware for the implementation of this project. The process followed to select the hardware is out of the scope of this paper. However, it is worth mentioning that the decisions related to hardware specifications were based on technical studies and evaluations. The main characteristics of the hardware dedicated to the cloud platform are:

- Servers: 75 blade servers (6 enclosures). Each blade has:
  - 128 GB of RAM
  - Dual 8 Core at 2,4 GHz
  - 2 146 GB SAS discs
- Storage appliance: the particular architecture of the storage system consists of three different layers, similar to the hierarchical memory model of any modern computer. These layers are composed by NL-SAS, SAS and SSD discs, which in total provide about 400 TB of RAW storage capacity.
- Network appliances: two dedicated advanced switches that offer a throughput of 24.3 Tbps per appliance, Software Defined Network (SDN) capabilities, layer 2 and layer 3 support for large deployments, and high availability, supported by their redundant architecture.

The selection of the CMP must be aligned with the hardware on which is going to be deployed; trying to take advantage of the specific characteristics that each component can provide. The final decision can also be substantiated on low level features, like hardware compatibility or integration with CMP.

At the software side, the University already has under its acquisitions some licenses for proprietary software, ranging from DBMS to virtualization platforms. The IT Department has deployed an Oracle database cluster, which is actively used by most of the University's software systems. In the case of virtualization software, the University has enough VMware vSphere licenses. These specific platforms should influence the selected cloud platform, in order to take advantage of already acquired software and know-how.

The following section describes the procedure to identify the available cloud platforms and how a subset of those solutions was chosen for the next step, the comparison of the software.

C. Preselection Process (step 3)

No matter how good the hardware is, its value depends on the software that manages it. Therefore, the selection of the software is crucial. Selecting the software that will run the cloud involves the research of available cloud platforms.

An initial review was done to determine the existing options and trends. We identified a variety of options. These options includes all the CMPs mentioned in the Related Work (Section II).

At this stage, we were not sure if all alternatives were suitable for our context, so, a filtering process was performed. We defined the following criteria needed to be met by UCR's cloud:

- CMP must provide IaaS model.
- Licensing model of the CMP must be open source: University's internal regulations dictate that the use of open source software should be preferred [35].
- CMP must be able to use our own hardware and infrastructure.
- CMP must be perceived as a solution with good evidence of usage and support: the access to documentation and case studies found for each of the platforms were used as parameters for this filter.

 ISBN: 978-1-61208-388-9

- Selected CMP must be robust and well known: external references to platform software and its use in industry (by large organizations) were factors used to measure the popularity of the product.
- Application scenario: the platform software must fit the specific purposes of the University. As stated before (Section IV-A), administrative/enterprise and academic/educational requirements must be met.

The preselected platforms were picked by the project research group and some of the IT Department's staff directly related with the project. At the end of the filtering process, the preselected CMPs were CloudStack, Eucalyptus, Openstack and VCL.

Openstack, CloudStack and Eucalyptus were chosen mainly because of the numerous evidence showing its use in enterprise scenarios. VCL is a CMP more focused on academic or educational areas, given the context and requirements on which the cloud is going to be deployed, this platform was of special interest for our evaluation.

Once the preselected platforms were established, we proceed to evaluate them using the comparison instrument that is going be described on the next section.

D. Comparison Criteria (step 4)

Based on the criteria presented by other comparisons [22][23][24][25][26][27][28], a preliminary set of criteria was defined. Some features or characteristics of concern were not present on other existing cloud platform comparisons. We added some new evaluation criteria according to our interests and evaluation purposes.

The criteria were grouped in two different categories: technical features and management features. Figure 3 shows detailed used criteria.

The technical features involve authentication mechanisms, networking, storage and other technical aspects. These features were also divided into eight categories. The general features involve the architecture of the platform, the supported DBMSs and the language in which it was developed. Authentication

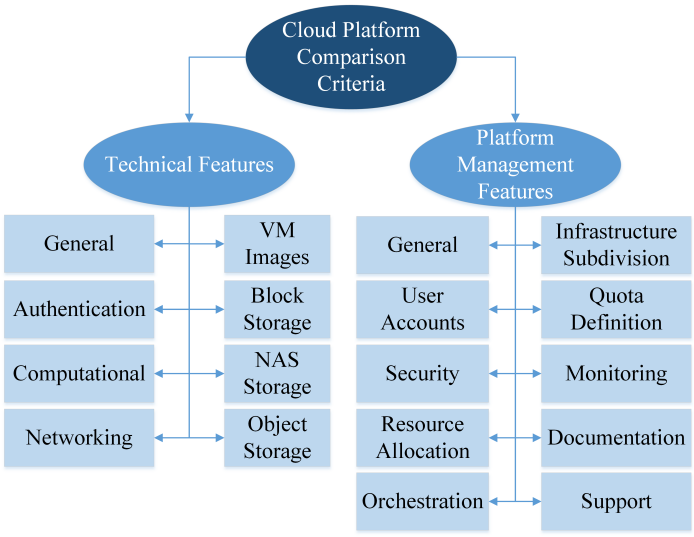


Figure 3. Criteria classification structure

features incorporate all security protocols. Computational features addressed the supported hypervisors and deployment techniques, and high performance computing support.

Networking features are related with IP version management, traffic isolation, and remote access capabilities. Virtual machine features were related with virtual machine images management and supported formats. Block, network attached and object storages evaluated all the available characteristics and actions for storage.

Management capabilities involve general features such as licensing, management interfaces, compatibility with other clouds through APIs and virtual machine administration. Management also involve user accounts and security features (privileges, roles and permissions).

Resource allocation, orchestration and infrastructural management features manage the distribution and scheduling of virtual machines and networking and storage components. They also managed the segregation of the infrastructure.

Quota definition establishes the usage limitation of resources for each user depending on roles and other aspects. Monitoring features involve the measurement of the cloud platform components. Finally, documentation and support features gives an idea of the usability and reliability of the evaluated cloud platforms.

E. Platforms Evaluation (step 5)

On the previous step a set of criteria was defined. The features selected for comparison were based on the parameters defined by other comparisons, and by our specific requirements. Once the comparison was made, a particular cloud platform was assigned to each of the project researchers. Each researcher was responsible for collecting the information of a specific platform. There was a cross validation between researchers to verify the gathered information. Every inconsistency was validated and resolved by the group.

The evaluation presented in [36], is a comparison table that shows which requirements are met and the main characteristics of each platform. The cloud software platforms included in the comparison were preselected on step 3 (Section IV-C), and the criteria used to evaluate those platforms were defined in step 4 (Section IV-D). The evaluation of characteristics was the last step of the procedure.

Tables II and III are segments of the complete CMPs comparison. Table II shows some of the comparison criteria related to technical features, and Table III refers to part of the evaluated management features. The extract in this paper only considered features in which Eucalyptus, OpenStack and CloudStack differ. The full version of the tables [36] shows not only all the criteria evaluated but also VCL is incorporated in the table.

With all the collected data (requirements, hardware specification and context variables) the final decision was made. The decision was made by the implementing team; composed by the seven researchers previously mentioned, the head of the IT Department, three Cloud Computing project leaders, and several collaborators (approximately 20 people).

The selection process described in Section IV let us define key factors to take into consideration. The evaluation of the cloud management platforms using the comparison tables

 ISBN: 978-1-61208-388-9

TABLE II. TECHNICAL FEATURES

General Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Version under revision	3.4.1	Havana, version 8 (2013.2.x)	4.2.0
Development language	Java, C, Python, Perl	Python	Java
Supported DBMS	PostGreSQL	Drizzle, Firebird, Microsoft SQL Server, MySQL, Oracle, PostgreSQL, SQLite, Sybase	MySQL
Supported protocols and backends	Local users, IAM	LDAP, Kerberos, Local Users	LDAP, local users
Computational Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Supported Hypervisors	ESXi, KVM	KVM, QEMU, Xen, ESXi/VC, Hyper-V, PowerVM, Docker	VMware, KVM, XenServer y Xen Cloud Platform
Scheduling methods	EuQoS	By filters and weights, random (filtered), customized	Not Found
Networking Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
IP v6 management	No	Not Found	No
Projects Traffic Isolation method	VLAN	VLAN, GRE, VLAN+GRE	VLAN
Remote Desktop access protocols	VNC	VNC (noVNC), SPICE	VNC
VM Images Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Supported disk formats	Raw, VMDK, VDI, QCOW2, ISO	Raw, VHD, VMDK, VDI, ISO, QCOW2, AKI, ARI, AMI	QCOW2, VHD, VMDK
Supported container formats	emi	Bare, ovf, aki, ari, ami	Not Found
Supported backends	Posix Filesystem	Local disc and NFS, Swift, S3, RBD	NFS, Swift, S3
Image caching	Yes	Yes	No
Create Image from VM snapshot	No	Yes	Yes
Block Storage Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Supported backends	NFS, DAS, EBS, iSCSI	NFS, LVM, Ceph, GlusterFS, and specific drivers (VNX, DELL, etc)	iSCSI, NFS
Clone a volume	Yes	Yes	No
Object Storage Features			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Backend	EBS, Walrus	Swift	Swift (OpenStack) and Amazon's S3
Access method	REST-Ful	REST-ful API & python-swiftclient	Just for VM Images and Volume Snapshots
Segmentation support (zones, groups)	No	Zones Rings Regions	Zones Rings Regions
Authentication integration support	GNU	Apache 2.0	Apache 2.0
Web UI integration	Yes	Yes	No

TABLE III. MANAGEMENT FEATURES

General			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Licensing	GNU	Apache 2.0	Apache 2.0
REST-ful API	Yes	Yes	No
User Accounts			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Permissions granularity	Groups, Users	Tenant, Users	Domain, Account, Domain Administrators, Projects
User with multiple projects or groups	No	Yes	Yes
Security			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Least privileged access design	Yes	Yes	Not Found
Fine granularity permissions definition method	JSON	JSON	On web UI
Centralized permission control	No	No	On web UI
Resource Allocation			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Virtual Machines block provisioning	Yes	Yes	Not Found
Orchestration			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Complex architectures definition support	Yes	Yes	No
Autoscaling support	No	Yes	Yes
Web UI integration	No	YEs	No
Architecture definition formats	Ansible	Heat Orchestration Template	No
Infrastructure Segregation			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Logic division	Availability Zones	Availability Zones >Host Aggregates	Zones >Pods >Clusters
Physical division	Regions	Cells and Regions	Regions
Quota Definition			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Volume quantity	Yes	Yes	Not Found
# of floating IPs	Yes	Yes	Not Found
# of security rules	Yes	Yes	Not Found
Monitoring			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
Web UI integration	No	Yes	Yes
Centralized module	No	Yes	Yes
Alarms definition support	No	Yes	Not Found
Computational resources monitoring	No	Yes	Yes
Networking resources monitoring	No	Yes	Yes
Images monitoring	No	Yes	No
Object Storage monitoring	No	Yes	No
Volumes monitoring	No	Yes	Yes
Energy monitoring	No	Yes	No
Visualizations included	No	Yes	Yes
Extendible (new metrics)	No	Yes	No
Documentation Support			
Feature	**Eucalyptus**	**OpenStack**	**CloudStack**
End user guide	Yes	Yes	Not Found
Support forum	Yes	Yes	No
Available certifications	Yes	Yes	No

 ISBN: 978-1-61208-388-9

made us realize that none of the preselected CMPs complied with all University's requirements. In the next section we describe how we addressed this issue.

V. Results and Discussion

The selected platforms were Openstack and VCL. Openstack was selected to fulfill the administrative requirements, and VCL to address the requirements associated to the academic area of the university. The decision to deploy two cloud platforms was taken primarily because none of them completely met the defined requirements (Section IV-A).

A characteristic of Openstack that stood out was the architecture design, specifically the method used for communication among its various components. The approach of a message queue as a general resource for inter-platform interaction was perceived as a key feature and a highly desired characteristic.

Also, cloud platform integration with our specific hardware was an important factor taken in account. Openstack block storage offers a driver to directly interact with our storage appliances (EMC VNXs).

Similarly, the SDN capability of Openstack matches the SDN support offered by our networking appliances. Openstack's Object Storage, known as Swift, offers scalability, which is required for the platform, given the amount of data to be stored. This is reflected on Figure 2, which indicates that backup service was one of the most required services by the end users. Swift is distributed, this means that data can be geographically distributed and replicated among the university's campuses.

Finally, Openstack has a solid base of community members, which enhances its perception as a top Cloud Operative System. This is evidenced by the contribution and involvement of big players such as Hewlett-Packard, Red Hat, Canonical, and others [14]. All the previously mentioned reasons made Openstack the selected platform to fulfill the administrative requirements.

For VCL, the main reason for its selection was the specific features that support the particularities of an academic scenario. This functionality was not supported by Openstack. As stated on Section IV-A, deployment of virtual labs was a mandatory functionality, this is covered by VCL. Also, VCL offers features for reservation of virtual machines based on given time and day. VCL has been already implemented in other universities; this is the case of North Carolina State University (where VCL was first implemented) [5]. We perceived this acceptance as an evidence of the compliance of this platform with academic requirements.

Even when the comparison tables provide an exhaustive description of each of the evaluated platforms, the final decision should not rely upon this as the final instrument. Instead, we suggest a conscious research and understanding of the preferred platforms. The main issue is that the tables do not provide an scaled evaluation, instead, they only provide a general overview that should be discussed to make a decision.

We presented a set of criteria used in order to select the best cloud platform for our scenario. These criteria could be used as a starting point for a cloud platform selection process, like the one presented in this paper. In [36], we show an in-depth vision of the platforms assessment. These comparison could be expanded/modified in order to suit specific interests and requirements depending on the implementation context.

The value of the presented process and comparison tables used is not the actual result of the selection process (the selection of Openstack and VCL), because it will certainly change depending on context and specific requirements. The real contribution of this research is the process and a set of criteria that could be used as a starting point to compare cloud platforms focused on the context.

VI. Conclusions and Future Work

We presented an implementation of a cloud platform to support administrative and academic requirements in an university setting. Our implementation was based on six steps followed to select the most appropriate cloud platform.

The first step, gathering requirements was addressed by performing a survey to final users, asking the IT Department for their administrative requirements and setting academic requirements with a panel of researchers. The second step helps to identify available resources (hardware and software), this step also allows to determine if new technologies are required. The third step we performed was the identification of all available platforms. Since the next steps in the selection process were intense and required lots of work, a basic filtering process was performed. We checked compliance with mandatory requirements: open source licensing model, implementation of IaaS and compatibility with the available hardware. Once we had a set of platforms that adapt to our mandatory requirements, a comparison between them allowed the selection of the best one(s). UCR's cloud uses OpenStack to support administrative requirements and VCL to support academia related requirements.

Every implementation of a cloud is very different. Thus, a standard definition of criteria is not possible; however, the criteria that we presented on [36] shows how we incorporated other works criteria and our own contextual requirements. The final steps are the evaluation of each possible platform with the criteria and the selection of the best one(s). We integrated various available CMPs comparisons available in literature and extended them for CloudStack, OpenStack, Eucalyptus and VCL.

Our future work consists of evaluating the process with other organizations to create a more robust process description and set of criteria. Other CMPs could be incorporated to the comparison tables to expand the selection possibilities. Taking into account that several comparisons are available in the literature but they widely differ in concepts granularity and completeness, we propose that a base criteria must be used as a comparison and any additional data should be added to the tables. If the tables presented in this research are expanded with more CMPs we could create a base line to compare CMPs based on a standard set of criteria.

Acknowledgment

We thank the IT Department at UCR for its collaboration on this research project. This work was supported by CITIC (*Centro de Investigaciones en Tecnologías de la Información y Comunicación*) at *Universidad de Costa Rica*, grand No. 834B3145 and by the *Escuela en Ciencias de la Computación e Informática* at *Universidad de Costa Rica*.

 ISBN: 978-1-61208-388-9

REFERENCES

[1] U. D. of Commerce, "National institute of standards and technology," 2014, URL: http://www.nist.gov/ [retrieved: 07, 2014].

[2] C. Baun, M. Kunze, J. Nimis, and S. Tai, Cloud Computing: Web-Based Dynamic IT Services, 1st ed. Springer Publishing Company, Incorporated, 2011.

[3] S. Kumar, S. Saravanan, and S. Mukherjee, "Recommendations for implementing cloud computing management platforms using open source," International Journal of Computer Engineering and Technology (IJCET), vol. 3, 2012.

[4] M. Vouk, "Cloud computing #2014; issues, research and implementations," in Information Technology Interfaces, 2008. ITI 2008. 30th International Conference on, June 2008, pp. 31–40.

[5] H. Schaffer, S. Averitt, M. Hoit, A. Peeler, E. Sills, and M. Vouk, "Ncsu's virtual computing lab: A cloud computing solution," Computer, vol. 42, no. 7, July 2009, pp. 94–97.

[6] D. Hamilton, "Cloud computing seen as next wave for technology investors," 2008, URL: http://www.financialpost.com/money/story.html?id=562877 [retrieved: 01, 2015].

[7] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, 2009, pp. 599 – 616.

[8] M. Caballer, D. Segrelles, G. Molto, and I. Blanquer, "A platform to deploy customized scientific virtual infrastructures on the cloud," in Science Gateways (IWSG), 2014 6th International Workshop on, June 2014, pp. 42–47.

[9] A. Inc., "Abiquo anycloud," 2014, URL: http://www.abiquo.com/ [retrieved: 07, 2014].

[10] A. S. Foundation, "Cloudstack," 2013, URL: http://cloudstack.apache.org/ [retrieved: 07, 2014].

[11] I. Eucalyptus Systems, "Eucalyptus," 2014, URL: https://www.eucalyptus.com/eucalyptus-cloud/iaas [retrieved: 07, 2014].

[12] U. of Chicago, "Nimbus," 2014, URL: http://www.nimbusproject.org/ [retrieved: 07, 2014].

[13] openQRM Enterprises, "openqrm," 2014, URL: http://www.openqrm-enterprise.com/ [retrieved: 07, 2014].

[14] R. C. Computing., "Openstack," 2014, URL: https://www.openstack.org/ [retrieved: 07, 2014].

[15] O. Project, "Opennebula project," 2002, URL: http://opennebula.org/ [retrieved: 07, 2014].

[16] Apache, "Vcl," 2012, URL: https://vcl.apache.org [retrieved: 07, 2014].

[17] L. Hewlett-Packard Development Company, "Hp helion cloudsystem," 2015, URL: http://www8.hp.com/us/en/cloud/cloudsystem.html [retrieved: 01, 2015].

[18] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system," in Cluster Computing and the Grid, 2009. CCGRID '09. 9th IEEE/ACM International Symposium on, May 2009, pp. 124–131.

[19] D. Dongre, G. Sharma, M. Kurhekar, U. Deshpande, R. Keskar, and M. Radke, "Scalable cloud deployment on commodity hardware using openstack," in Advanced Computing, Networking and Informatics-Volume 2, ser. Smart Innovation, Systems and Technologies, M. Kumar Kundu, D. P. Mohapatra, A. Konar, and A. Chakraborty, Eds., vol. 28. Springer International Publishing, 2014, pp. 415–424.

[20] J. Vilaplana, F. Solsona, J. Mateo, and I. Teixido, "Sla-aware load balancing in a web-based cloud system over openstack," in Service-Oriented Computing – ICSOC 2013 Workshops, ser. Lecture Notes in Computer Science, A. Lomuscio, S. Nepal, F. Patrizi, B. Benatallah, and I. Brandić, Eds., vol. 8377. Springer International Publishing, 2014, pp. 281–293.

[21] S. Saibharath and G. Geethakumari, "Design and implementation of a forensic framework for cloud in openstack cloud platform," in Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on, Sept 2014, pp. 645–650.

[22] T. Cordeiro, D. Damalio, N. Pereira, P. Endo, A. Palhares, G. Gonçalves, D. Sadok, J. Kelner, B. Melander, V. Souza, and J.-E. Mångs, "Open source cloud computing platforms," in Grid and Cooperative Computing (GCC), 2010 9th International Conference on, Nov 2010, pp. 366–371.

[23] I. Voras, B. Mihaljevic, and M. Orlic, "Criteria for evaluation of open source cloud computing solutions," in Information Technology Interfaces (ITI), Proceedings of the ITI 2011 33rd International Conference on, June 2011, pp. 137–142.

[24] I. Voras, B. Mihaljevic, M. Orlic, M. Pletikosa, M. Zagar, T. Pavic, K. Zimmer, I. Cavrak, V. Paunovic, I. Bosnic, and S. Tomic, "Evaluating open-source cloud computing solutions," in MIPRO, 2011 Proceedings of the 34th International Convention, May 2011, pp. 209–214.

[25] S. Wind, "Open source cloud computing management platforms: Introduction, comparison, and recommendations for implementation," in Open Systems (ICOS), 2011 IEEE Conference on, Sept 2011, pp. 175–179.

[26] D. Cerbelaud, S. Garg, and J. Huylebroeck, "Opening the clouds: Qualitative overview of the state-of-the-art open source vm-based cloud management platforms," in Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware, ser. Middleware '09. New York, NY, USA: Springer-Verlag New York, Inc., 2009, pp. 22:1–22:8.

[27] N. Khan, A. Noraziah, T. Herawan, and M. Deris, "Cloud computing: Analysis of various services," in Information Computing and Applications, ser. Lecture Notes in Computer Science, B. Liu, M. Ma, and J. Chang, Eds., vol. 7473. Springer Berlin Heidelberg, 2012, pp. 397–404.

[28] A. Li, X. Yang, S. Kandula, and M. Zhang, "Cloudcmp: Comparing public cloud providers," in Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, ser. IMC '10. New York, NY, USA: ACM, 2010, pp. 1–14.

[29] U. de Costa Rica, "Oplau," 2014, URL: http://oplau.ucr.ac.cr/ [retrieved: 07, 2014].

[30] A. Kuo, "Opportunities and challenges of cloud computing to improve health care services," J Med Internet Res, vol. 13, no. 3, Sep 2011, p. e67.

[31] CINDE, "Services sector," 2015, URL: http://www.cinde.org/en/sectors/services [retrieved: 01, 2015].

[32] S. Averitt, M. Bugaev, A. Peeler, H. Shaffer, E. Sills, S. Stein, J. Thompson, and M. Vouk, "Virtual computing laboratory (vcl)," in Proceedings of the International Conference on Virtual Computing Initiative. IBM, 2007, pp. 1–16.

[33] M. Rouse, "Bare-metal provisioning," 2014, URL: http://searchdatacenter.techtarget.com/definition/bare-metal-provisioning [retrieved: 10, 2014].

[34] R. T. Fielding and R. N. Taylor, "Principled design of the modern web architecture," in Proceedings of the 22Nd International Conference on Software Engineering, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 407–416. [Online]. Available: http://doi.acm.org/10.1145/337180.337228

[35] U. de Costa Rica, "Centro de informática, normativas," 2012, URL: http://ci.ucr.ac.cr/node/105 [retrieved: 07, 2014].

[36] C. de Investigaciones en Tecnologías de la Información y Comunicación, "Cloud platforms comparison: a technical report," 2014, URL: http://citic.ucr.ac.cr/cloud-platform-comparison-technical-report/ [retrieved: 10, 2014].

 ISBN: 978-1-61208-388-9

Taxonomy of Deployment Patterns for Cloud-hosted Applications: A Case Study of Global Software Development (GSD) Tools

Laud Charles Ochei, Julian M. Bass, Andrei Petrovski

School of Computing Science and Digital Media
Robert Gordon University
Aberdeen, United Kingdom
Emails: {l.c.ochei,j.m.bass,a.petrovski}@rgu.ac.uk

***Abstract*—Cloud patterns describe deployment and use of various cloud-hosted applications. There is little research which focuses on applying these patterns to cloud-hosted Global Software Development (GSD) tools. As a result, it is difficult to know the applicable deployment patterns, supporting technologies and trade-offs to consider for specific software development processes. This paper presents a taxonomy of deployment patterns for cloud-hosted applications. The taxonomy is composed of 24 sub-categories which were systematically integrated and structured into 8 high-level categories. The taxonomy is applied to a selected set of software tools: JIRA, VersionOne, Hudson, Subversion and Bugzilla. The study confirms that most deployment patterns are related and cannot be fully implemented without being combined with others. The taxonomy revealed that (i) the functionality provided by most deployment patterns can often be accessed through an API or plugin integrated with the GSD tool, and (ii) RESTful web services and messaging are the dominant strategies used by GSD tools to maintain state and exchange information asynchronously, respectively. We also provide recommendations to guide architects in selecting applicable deployment patterns for cloud deployment of GSD tools.**

Keywords–Taxonomy; Deployment Pattern; Cloud-hosted Application; GSD Tool; Plugin.

I. INTRODUCTION

Collaboration tools that support Global Software Development (GSD) processes are increasingly being deployed on the cloud [1][2]. The architectures/patterns used to deploy these tools to the cloud are of great importance to software architects because it determines whether or not the system's required quality attributes (e.g., performance) will be exhibited [3][4][5].

Collections of cloud patterns exist for describing the cloud, and how to deploy and use various cloud offerings [6][7]. However, there is little or no research in applying these patterns to describe the cloud-specific properties of applications in software engineering domain (e.g., collaboration tools for GSD, hereafter referred to as GSD tools) and the trade-offs to consider during cloud deployment. This makes it very challenging to know the deployment patterns (together with the technologies) required for deploying GSD tools to the cloud to support specific software development processes (e.g., continuous integration (CI) of code files with Hudson).

Motivated by this problem, we propose a taxonomy of deployment patterns for cloud-hosted applications to help software architects in selecting applicable deployment patterns for deploying GSD tools to the cloud. We are inspired by the work of Fehling et al. [6], who catalogued a collection of patterns that will help architects to build and manage cloud applications. However, these patterns were not applied to any specific application domain, such as cloud-hosted GSD tools.

The research question this paper addresses is: **"How can we create and use a taxonomy for selecting applicable deployment patterns for cloud deployment of GSD tools."** It is becoming a common practice for distributed enterprises to hire cloud deployment architects or "application deployers" to deploy and manage cloud-hosted GSD tools [8]. Saleforce's Continuous Integration systems, for example, runs 150000 + test in parallel across many servers and if it fails it automatically opens a bug report for developers responsible for that *checkin* [9].

We created and applied the taxonomy against a selected set of GSD tools derived from an empirical study [10] of geographically distributed enterprise software development projects. The overarching result of the study is that most deployment patterns are related and have to be combined with others during implementation, for example, to address hybrid deployment scenarios, which usually involves integrating processes and data in multiple clouds.

The main contributions of this paper are:
1. Creating a novel taxonomy of deployment patterns for cloud-hosted application.
2. Demonstrating the practicality of the taxonomy by applying it to: (i) position a selected set of GSD tools; and (ii) compare the cloud deployment requirements of GSD tools.
3. Presenting recommendations and best practice guidelines for identifying applicable deployment patterns together with the technologies for supporting cloud deployment of GSD tools.

The rest of the paper is organized as follows: Section 2 gives an overview of the basic concepts related to deployment patterns for Cloud-hosted GSD tools. In Section 3, we discuss the research methodology including taxonomy development, tools selection, application and validation. Section 4 presents the findings of the study focusing on positioning a set of GSD tools within the taxonomy. In Section 5, we discuss the lessons learned from applying the taxonomy. The recommendations and limitations of the study are in Sections 6 and 7 respectively. Section 8 is reporting the conclusion and future work.

 ISBN: 978-1-61208-388-9

II. Deployment Patterns for Cloud-hosted GSD Tools

A. Global Software Development

Definition 1: **Global Software Development.** GSD is defined by Lanubile [11] as the splitting of the development of the same software product or service among globally distributed sites. Since there are many stakeholders in GSD, there is need to have tools that support collaboration and integration among the team members involved in software development [12]. The work of Portillo et al. [13] focused on categorizing various tools used for collaboration and coordination in Global Software Development.

Definition 2: **Cloud-hosted GSD Tool.** "Cloud-hosted GSD Tool" refers to collaboration tools used to support GSD processes in a cloud environment. We adopt the: (i) NIST Definition of Cloud Computing to define properties of cloud-hosted GSD tools; and (ii) ISO/IEC 12207 as our classification frame for defining the scope of a GSD tool. Portillo et al. [13] identified three groups of GSD tools for supporting ISO/IEC 12207 processes - tools to support Project Processes (e.g., JIRA), tools to support Implementation Processes such as requirements analysis, integration process (e.g., Hudson) and tools for Support Processes (e.g., Subversion). These GSD tools, also referred to as Collaboration tools for GSD [13], are increasingly being deployed to the cloud for Global Software Development by large distributed enterprises.

B. Cloud Deployment Patterns

Definition 3: **Cloud Deployment Pattern.** We define a "Cloud deployment pattern" as a type of architectural pattern which embodies decisions as to how elements of the cloud application will be assigned to the cloud environment where the application is executed. Architectural and design patterns have long been used to provide known solutions to a number of common problems facing a distributed system [4][14].

Our definition of cloud deployment pattern is similar to the concept of design patterns [14], (architectural) deployment patterns [4], collaboration architectures [3], cloud computing patterns [6], cloud architecture patterns [15], and cloud design patterns [7]. These concepts serve the same purpose in the cloud (as in many other distributed environments). For example, the generic architectural patterns- client-server, peer-to-peer, and hybrid [4] - relates to the following: (i) the 3 main collaboration architectures, i.e., centralized, replicated and hybrid [3]; and (ii) cloud deployment patterns -2-tier, content distribution network and hybrid data [6].

C. Taxonomy of Cloud Computing Patterns

Taxonomies and classifications facilitate systematic structuring of complex information. In software engineering, they are used for comparative studies involving tools and methods, for example, software evolution [16] and Global Software Engineering [17]. In this paper, we focus on using a taxonomy to structure cloud deployment patterns for cloud-hosted applications, in particular in the area of GSD tools.

Several attempts have been made by researchers to create classifications of cloud patterns to build, and deploy cloud-based applications. For example, Wilder [15] describes eleven patterns and then illustrates with the Page of Photos application and Windows Azure how each pattern can be used to build cloud-native applications. A collection of over 75 patterns (with known uses of their implementation) for building and managing a cloud-native application are provided by Fehling et al. [6]. Homer et al. [7] describes 24 patterns, for solving common problems in cloud application development. Moyer [18] also documents a collection of patterns and then uses a simple Weblog application to illustrate the use of these patterns. Other documentation of cloud deployment patterns can be found in [19][20][21][22].

Cloud patterns in existing classifications are applied to simple web-based applications (e.g., Weblog application [18]) without considering the different application processes they support. Moreover, these patterns have not been applied against a set of applications in software engineering domain, such as cloud-hosted GSD tools. GSD tools may have similar architectural structure but they (i) support different software development processes, and (ii) impose varying workload on the cloud infrastructure, which would influence the choice of a deployment pattern. ***Motivated by these shortcomings, we extend the current research by developing a taxonomy of deployment patterns for cloud-hosted applications and applying it to a set of GSD tools.***

III. Methodology

A. Development of the Taxonomy

We develop the taxonomy using a modified form of the approach used by Lilien [23] in his work for building a taxonomy of specialized ad hoc networks and systems for a given target application class. The approach summaries to the following steps:

Step 1: Select the target class of Software Tool- The target class is based on the ISO/IEC 12207 taxonomy for software life cycle process (see Definition 3 for details). The following class of tools are excluded: (i) tools not deployed in a cloud environment (even if they are deployed on a dedicated server to perform the same function); and (ii) general collaboration tools and development environments (e.g., MS Word, Eclipse).

Step 2: Determine the requirements for the Taxonomy- The first requirement is that the taxonomy should incorporate features that restricts it to GSD tools and Cloud Computing. In this case, we adopt the ISO/IEC 12207 framework [13] and NIST cloud computing definition [24]. Secondly, it should capture the components of an (architectural) deployment structure [4] - software elements (i.e., GSD tool to be deployed) and external environment (i.e., cloud environment). Therefore our proposed taxonomy is a combination of two taxonomies - Taxonomy A, which relates to the components of the cloud environment [24], and Taxonomy B which relates to the components of the cloud application architecture [6].

Step 3: Determine and prioritize the set of all acceptable categories and sub-categories of the Taxonomy- Determine and prioritize the set of all acceptable categories and sub-categories of the Taxonomy- We prioritized the categories of the taxonomy to reflect the structure of a cloud stack from physical infrastructure to the software process of the deployed GSD tool. The categories and sub-categories of the 2 taxonomies are described as follows:

(1) Application Process: the sub-categories (i.e., project processes, implementation processes and support processes) represent patterns for handling the workload imposed on the cloud infrastructure by the ISO/IEC 12207 software processes supported by GSD tools [13]; *(2) Core cloud properties:* the

 ISBN: 978-1-61208-388-9

sub-categories (i.e., rapid elasticity, resource pooling and measured service) contain patterns used to mitigate the core cloud computing properties of the GSD tools [6]; *(3) Service Model:* the sub-categories reflect cloud service models- SaaS, PaaS, IaaS [24]; *(4) Deployment Model:* the sub-categories reflect cloud deployment models- private, community, public and hybrid [24]; *(5) Application Architecture:* the sub-categories represent the architectural components that support a cloud-application such as application components (e.g., presentation, processing, and data access), multitenancy, and integration [6]; *(6) Cloud Offerings:* the sub-categories reflect the major infrastructure cloud offerings that can be accessed- cloud environment, processing, storage and communication offering [6]; *(7) Cloud Management:* contains patterns used to manage both the components and processes/runtime challenges) of GSD tools. The 2 sub-categories are - management components and processes [6]; *(8) Composite Cloud:* contains compound pattern (i.e., patterns that can be formed by combining other patterns or can be decomposed into separate components). The sub-categories are: decomposition style and hybrid cloud application [6].

Step 4: Determine the space of the Taxonomy- The selected categories and their associated sub-categories define the space of the taxonomy. Table 1 show the taxonomy captured in one piece. The upper-half represents Taxonomy A which is based on NIST Cloud Computing Definition, while the lower-half represents Taxonomy B which is based on the components of a typical cloud application architecture. Each Taxonomy, A and B, has four categories, each with a set of sub-categories. Entries in the "Related Pattern" column show examples of patterns drawn from well-known collections of cloud patterns such as [6][15][7]. The thick lines (Table I) show the space occupied by patterns used for hybrid-deployment scenarios.

TABLE I. TAXONOMY OF DEPLOYMENT PATTERNS FOR CLOUD-HOSTED APPLICATIONS

Deployment Components	**Categories of Deployment Patterns**		**Related Patterns**
	Main Categories	*Sub-Categories*	
Cloud-hosted Environment (Taxonomy A)	Application Process	Project processes	Static workload
		Implementation processes	Continuously changing workload
		Support processes	Continuously changing workload
	Core Cloud Properties	Rapid Elasticity	Elastic platform,Autoscaling[15]
		Resource Pooling	Shared component, Private cloud
		Measured Service	Elastic Platform, Throttling[7]
	Cloud Service Model	Software resources	SaaS
		Platform resources	PaaS
		Infrastructure resources	IaaS
	Cloud Deployment Model	Private clouds	Private cloud
		Community clouds	Community cloud
		Public clouds	Public cloud
		Hybrid clouds	Hybrid cloud
	Composite Cloud Application	Hybrid cloud applications	Hybrid Processing, Hybrid Data,Multisite Deployment [15]
Cloud-hosted Application (Taxonomy B)		Decomposition style	2-tier/3-tier application, Content Delivery Network [15]
	Cloud Management	Management Processes	Update Transition Process, Scheduler Agent [7]
		Management Components	Elastic Manager, Provider Adapter, External Configuration Store [7]
	Cloud Offerings	Communication Offering	Virtual Networking, Message-Oriented Middleware
		Storage Offering	Block Storage, Database Sharding [15], Valet Key [7]
		Processing Offerings	Hypervisor, Map Reduce [15]
		Cloud Environment Offerings	Elastic Infrastructure, Elastic Platform, Runtime Reconfiguration [7]
	Cloud Application Architecture	Integration	Integration Provider, Restricted Data Access Component
		Multi-tenancy	Shared Component, Tenant-Isolated Component
		Application components	Stateless Component, User Interface Component

TABLE II. PARTICIPATING COMPANIES, SOFTWARE PROJECTS, SOFTWARE-SPECIFIC PROCESS AND GSD TOOLS USED

Companies	Projects	Software process	GSD tool
Company A, Bangalore	Web Mail Web Calendar	Issue tracking Code integration	JIRA Hudson
Company B, Bangalore	Web Mail Web Calendar	Issue tracking Version control	JIRA Subversion
Company H, Delhi	Customer service Airline	Agile tailoring Issue tracking	VersionOne JIRA
Company D, Bangalore (Offshore Provider to Company E)	Marketing CRM	version control Error tracking	Subversion Bugzilla
Company E, London	Banking Marketing CRM	Issue tracking Agile tailoring Code Building	JIRA VersionOne Hudson

B. GSD Tool Selection

We carried out an empirical study to find out: (1) the type of GSD tools used in large-scale distributed enterprise software development projects; and (2) what tasks they utilize the GSD tools for.

1) Research Site: The study involved 8 international companies and interviews were conducted with 46 practitioners. The study was conducted between January, 2010 and May, 2012; and then updated between December, 2013 and April, 2014. The companies were selected from a population of large enterprises involved in both on-shore and off-shore software development projects. The companies had head offices in countries spread across three continents: Europe (UK), Asia (India), and North America (USA). Data collection involved document examination/reviews, site visits, and interviews. Further details of the data collection and data analysis procedure used in the empirical study can be seen in Bass [10].

2) Derived Dataset of GSD Tools: The selected set of GSD tools are: JIRA [25], VersionOne [26], Hudson [27], Subversion [28] and Bugzilla [29]. We selected these tools for two main reasons: (i) Practitioners confirmed the use of these tools in large scale geographically distributed enterprise software development projects [10]; (ii) The tools represents a mixture of open-source and commercial tools that support different software development processes; and are associated with stable developers community (e.g., Mozilla Foundation) and publicly available records (e.g., developer's websites, whitepapers, manuals). Table 2 (another view of the one in [10]) shows the participating companies, projects and the GSD tools they used.

C. Applying the Taxonomy

We demonstrate the practicality of the taxonomy by applying it to position a selected set of GSD tools. We used the collection of patterns from [6] as our reference point, and then complimented the process with patterns from [15][7].

The structure of the positioned deployment pattern, in its textual form, is specified as a string consisting of three sections-(i) Applicable deployment patterns; (ii) Technologies required to support such implementation; and (iii) Known uses of how the GSD tool (or one of its products) implements or supports the implementation of the pattern. In a more general sense, the string can be represented as: [PATTERN; TECHNOLOGY; KNOWN USE]. When more than one pattern or technology is applicable, we separate them with commas. Each sub-category of the taxonomy represents a unique class of reoccurring cloud deployment problem, while the applicable deployment pattern represents the solution.

 ISBN: 978-1-61208-388-9

D. Validation of the Taxonomy

We validate the taxonomy in theory by adopting the approach used by Smite et al. [17] to validate his proposed taxonomy for terminologies in global software engineering. A taxonomy can be validated with respect to completeness by benchmarking against existing classifications and demonstrating its utility to classify existing knowledge [17].

We have benchmarked Taxonomy A to existing classifications: the ISO/IEC 12207 taxonomy of software life cycle processes and the components of a cloud model based on NIST cloud computing definition, NIST SP 800-145. Taxonomy B is benchmarked to components of a cloud application architecture such as cloud offering and cloud management, as proposed by Fehling et al. [6]. The collection of patterns in [6] captures all the major components/processes required to support a typical cloud-based application, such as cloud management and integration.

We demonstrate the utility of our taxonomy by positioning the 5 selected GSD tools within the taxonomy to evaluate applicable deployment patterns together with the supporting technologies for deploying GSD tools to the cloud. Table 3 and 4 show that several deployment patterns (selected from 4 studies) can be placed in the sub-categories of our taxonomy.

IV. FINDINGS

In this section, we present the findings obtained by applying the taxonomy against a selected set of GSD tools: JIRA, VersionOne, Hudson, Subversion and Bugzilla. Refer to section III- B for details of the processes supported by these tools.

A. Comparing the two Taxonomies

The cloud deployment patterns featured in Taxonomy A (i.e., upper part of Table 1) relates to the cloud environment hosting the application, while the cloud deployment patterns in Taxonomy B (i.e., lower part of Table 1) relates to the cloud-hosted application itself. For example, the PaaS pattern is used to provide an execution environment to customers on the provider-supplied cloud environment. Elastic platform pattern can be used in the form of a middleware integrated into a cloud-hosted application to provide an execution environment.

B. Hybrid-related deployment Patterns

Both taxonomies contain patterns for addressing hybrid deployment scenarios (i.e., the space demarcated with thick lines). For example, a hybrid cloud (Taxonomy A) integrates different clouds and static data centers to form a homogeneous hosting environment, while hybrid data (Taxonomy B) can be used in a scenario where data of varying sizes generated from a GSD tool resides in an elastic cloud and the remainder of the application resides in a static environment.

C. Patterns for Implementing Elasticity

We have observed patterns that can be used by GSD tools to address rapid elasticity at all levels of the cloud stack. For example, Elastic manager can be used at the application level. Elastic platform and Elastic infrastructure can be used at the platform, and Infrastructure resources level, respectively.

D. Positioning of GSD tools on the Taxonomy

Tables 2 and 3 show the findings obtained by positioning the cloud-hosted GSD tools on each sub-category of the taxonomy. In the following, we present a shortlist of these findings to show that we can identify applicable deployment patterns to address a wide variety of deployment problems.

(i) All the GSD tools considered in this study are based on web-based architecture. For example, Bugzilla and JIRA are designed as a web-based application, which allows for separation of the user interface, and processing layers from the database that stores details of bugs/issues being tracked.

(ii) All the GSD tools have support for API/Plugin architecture. For example, JIRA supports several API's that allows it to be integrated with other GSD tools. Bugzilla:Web services, a standard API for external programs to interact with Bugzilla implies support for stateless pattern. These APIs represent known uses how these deployment patterns are implemented.

(iii) Virtualization is a key supporting technology used in combination with other patterns to achieve elasticity at all levels of the cloud stack, particularly in ensuring fast provisioning and de-provisioning of infrastructure resources.

(iv) The GSD tools use Web services (through a REST API in patterns such as integration provider [6]) to hold external state information, while messaging technology (through message queues in patterns such as Queue-centric workflow [15] and Queue-based load leveling [7]) is used to exchange information asynchronously between GSD tools/components.

(vi) Newer commercial GSD tools (JIRA and VersionOne) are directly offered as SaaS on the public cloud. On the other hand, older open-source GSD tools (Hudson, Subversion and Bugzilla) are the preferred for private cloud deployment. They are also available on the public cloud, but by third party cloud providers.

We summarize our findings as follows: Although there are a few patterns that are mutually exclusive (e.g., stateless and statefull components [6]), most patterns still have to be combined with others (e.g, combining PaaS with Elastic platform). These deployment patterns may also use similar technologies such as REST, messaging and virtualization to facilitate their implementation.

V. DISCUSSION

The findings clearly suggest that by positioning a set of GSD tools on our proposed taxonomy, the purpose of the study has been achieved. The overarching result of the study is that most deployment patterns have to be combined with others during implementation. The findings presented here support previous research suggesting that most patterns are related and so two or more patterns can be used together [4][14].

A. Combining Related Deployment Patterns

Many deployment patterns are related and cannot be fully implemented without being combined with other ones, especially to address hybrid deployment scenarios. This scenario is very common in collaborative GSD projects, where a GSD

 ISBN: 978-1-61208-388-9

TABLE III. POSITIONING GSD TOOLS ON THE PROPOSED TAXONOMY (TAXONOMY A)

Category	Sub-Category	*JIRA*	*VersionOne*	*Hudson*	*Subversion*	*Bugzilla*
Application Process	Project processes	**Static workload, Continuously changing workload**; *SaaS*; JIRA used by small no. of users, issues tracked reduces over time[25]	**Static workload**; *SaaS*; VersionOne is installed for a small number of users[26]	Process not supported	Process not supported	Process not supported
	Implementation processes	Process not supported	Process not supported	**Continuously changing workload**; *PaaS*; Hudson builds reduces gradually as project stabilizes)[27]	Process not supported	Process not supported
	Support processes	Process not supported	Process not supported	Process not supported	**Static workload, Continuously changing workload**;*PaaS, Hypervisor*; rate of code files checked into Subversion repository is nearly constant or reduces over time[28]	**Continuously changing workload**; *PaaS,Hypervisor*; Errors tracked using Bugzilla reduces over time[29]
Core Cloud Properties	Rapid Elasticity	**Stateless pattern, Elastic platform**; *REST API*; JIRA is installed in cloud as SaaS[25]	**Stateless pattern, Elastic platform**; *REST API*; VersionOne is installed in cloud as SaaS[26]	**Elastic infrastructure, shared component**; *hypervisor*; Hudson server is supported by hypervisor in a private cloud[27]	**Elastic infrastructure, tenant-isolation component**; *hypervisor*; Subversion repository is supported by Elastic infrastructure[28]	**Stateless pattern**; *REST API*; Bugzilla is installed in cloud as SaaS in private cloud[29]
	Resource Pooling	**Hypervisor, Public Cloud,** ; *Virtualization*; JIRA deployed on the public cloud as SaaS[25]	**Hypervisor, Public cloud**; *Virtualization*; VersionOne deployed on public cloud as SaaS[26]	**Hypervisor, Tenant-isolated component**; *Virtualization*; Hudson is deployed on a hypervisor[27]	**Hypervisor, Tenant-isolated component**; *Virtualization*; Subversion is deployed on a hypervisor[25]	**Hypervisor, Public cloud**; *Virtualization*; Bugzilla deployed on the public cloud[29]
	Measured Service	**Static workload, Elastic Infrastructure,Throttling[7]**; *Virtualization*; Small number JIRA users generates a nearly constant workload[25]	**Static workload, Elastic Infrastructure,Throttling[7]**; *Virtualization*; Small number of VersionOne users generates small workload[26]	**Static workload, Elastic Infrastructure,Throttling[7]**; *Virtualization*; Hudson can be supported on public cloud by elastic infrastructure[27]	**Static workload, Elastic Infrastructure,Throttling[7]**; *Virtualization*; Subversion can be supported on public cloud by elastic infrastructure[28]	**Static workload, Elastic Infrastructure,Throttling[7]**; *Virtualization*; Bugzilla can be supported on third party public cloud by elastic infrastructure[29]
Cloud Service Model	Software resources	**SaaS**; *Web Services, REST*; JIRA OnDemand[25]	**SaaS**; *Web Services, REST*; VersionOne OnDemand[26]	**SaaS**; *Web Services, REST*; Hudson is offered by 3^{rd} party cloud providers like CollabNet[30]	**SaaS**; *Web Services, REST*; Subversion is offered by 3^{rd} party cloud providers like CollabNet[30]	**SaaS**; *Web Services, REST*; Bugzilla is offered by 3^{rd} party cloud providers like CollabNet[30]
	Platform resources	**PaaS**; *Elastic platform, Message Queuing*; JIRA Elastic Bamboo[25]	**PaaS**; *Elastic platform, Message Queuing*; No known use	**PaaS**; *Elastic platform, Message Queuing*; Build Doctor and Amazon EC2 for Hudson	**PaaS**; *Elastic platform, Message Queuing*; Flow Engine powered by Jelastic for Subversion	**PaaS**; *Elastic platform, Message Queuing*; No known use
	Infrastructure resources	Not applicable	Not applicable	**IaaS**; *Hypervisor*; Hudson is a distributed execution system comprising master/slave servers[27]	**IaaS**; *Hypervisor*; Subversion can be deployed on a hypervisor	Not applicable
Cloud Deployment Model	Private usage	**Private cloud**; *Hypervisor*; JIRA can be deployed on private cloud using private cloud software like OpenStack	**Private cloud**; *Hypervisor*; VersionOne On-premises[26]	**Private cloud**; *Hypervisor*; Hudson can be deployed on private cloud using private cloud software	**Private cloud**; *Hypervisor*; Subversion can be deployed on private cloud using private cloud software	**Private cloud**; *Hypervisor*; Bugzilla can be deployed on private cloud using private cloud software
	Community usage	**Community cloud**; *SaaS*; Bugzilla can be deployed on private cloud	**Community cloud**; *SaaS*; Bugzilla can be deployed on community cloud	**Community cloud**; *SaaS,Paas, IaaS*; Bugzilla can be deployed on community cloud	**Community cloud**; *SaaS,IaaS*; Bugzilla can be deployed on community cloud	**Community cloud**; *SaaS, PaaS*; Bugzilla can be deployed on community cloud
	Public usage	**Public cloud**; *SaaS*; JIRA OnDemand is hosted on public cloud[25]	**Public cloud**; *SaaS*; VersionOne is hosted on public cloud[26]	**Public cloud**; *SaaS,PaaS,IaaS*; Hudson is hosted on public cloud(via 3^{rd} party providers)[30]	**Public cloud**; *SaaS, IaaS*; Subversion is hosted on public cloud(via 3^{rd} party providers)[30]	**Public cloud**; *SaaS, PaaS*; Bugzilla is hosted on public cloud(via 3^{rd} party providers)[30]
	Hybrid usage	**Hybrid cloud**; *SaaS*; JIRA used to track issues on multiple clouds	**Hybrid cloud**; *SaaS*; Agile projects are stored in different clouds[28]	**Hybrid cloud**; *SaaS,PaaS, IaaS*; Hudson builds done in separate cloud	**Hybrid cloud**; *SaaS, IaaS*; Subversion repository resides in multiple clouds	**Hybrid cloud**; *SaaS, PaaS*;Bugzilla DB can be stored in different clouds

tool either requires multiple cloud deployment environments or components, each with its own set of requirements. Our taxonomy, unlike others [15][7], clearly shows where to look for hybrid-related deployment patterns (i.e., the space demarcated by thick lines in Table I) to address this challenge. For example, when using Hudson there is usually a need to periodically extract the data its generates to store in an external storage during continuous integration of files. This implies the implementation of a hybrid data pattern. Hudson can be used in combination with other GSD tools such as Subversion (for version control) and Bugzilla (for error tracking) within a particular software development project, each of which may also have their own deployment requirements.

B. GSD Tool Comparison

The taxonomy gives us a better understanding of various GSD tools and their cloud specific features. While other taxonomies and classifications use simple web applications [15] to exemplify their patterns, we use a mixture of commercial and open-source GSD tools. For example, commercial GSD tools (i.e., JIRA and VersionOne) are offered as a SaaS on the public cloud and also have a better chance of reflecting the essential cloud characteristic. Their development almost coincides with the emergence of cloud computing, allowing new features to be introduced into revised versions. The downside is that they offer less flexibility in terms of customization [31]. On the other hand, open-source GSD tools (i.e., Hudson, Subversion) are provided on the public cloud by third party providers and they rely on API/plugins to incorporate support for most cloud features. The downside is that many of the plugins available for integration are not maintained by the developer's community and so consumers use them at their own risk. The taxonomy also revealed that open-source GSD tools (e.g., Hudson, Subversion) are used at a later stage of a software life-cycle process in contrast to commercial tools which are used at the early stages.

C. Supporting Technologies and API/Plugin Architecture

Another interesting feature of our taxonomy is that by positioning the selected GSD tools on it, we discovered that the

 ISBN: 978-1-61208-388-9

TABLE IV. POSITIONING GSD TOOLS ON THE PROPOSED TAXONOMY (TAXONOMY B).

Category	Sub-Category	*JIRA*	*VersionOne*	*Hudson*	*Subversion*	*Bugzilla*
Application Architecture	Application Components	**User interface component,Stateless**; *REST API, AJAX*; State information in JIRA thru REST API[25]	**User-interface component,Stateless**; *jQuery AJAX, REST/Web Service*; VersionOne REST API[26]	**User-interface component,Stateless**; *REST API, AJAX*; Hudson Dashboard pages via REST[27]	**User-interface component,Stateless**;*REST API, AJAX*; ReSTful Web Services used to interact with Subversion Repositories [28]	**Stateless**; *Bugzilla:WebService API*; Bugzilla::WebService API[29]
	Multitenancy	**Shared component**; *Elastic Platform, Hypervisor*; JIRA login system[25]	**Shared component**; *Hypervisor*; VersionOne supports re-useable configuration schemes[26]	**Shared component**; *Hypervisor*; Hudson 3.2.0 supports multi-tenancy with Job Group View and Slave isolation[27]	**Tenant Isolated component**; *Hypervisor*; Global search/replace operations are shielded from corrupting subversion repository.[28]	**Shared component**; *Hypervisor*; Different users are virtually isolated within Bugzilla DB[29]
	Cloud Integration	**Restricted Data Access component, Integration provider**; *REST API*; JIRA REST API is used to integrate JIRA with other applications[25]	**Integration provider**; *REST, Web Services*; VersionOne OpenAgile Integrations platform, REST Data API for user stories[26]	**Integration provider**; *REST, Web Services*; Stapler component of Hudson's architecture uses REST[27]	**Integration provider**; *REST, Web Services*; Subversion API[28]	**Integration provider**; *REST, Web Services*; Bugzilla::WebService API[29]
Cloud Offering	Cloud environment Offering	**Elastic platform**; *PaaS*; JIRA Elastic Bamboo runs builds to create instances of remote agents in the Amazon EC2[25]	**Integration provider**; *REST, Web Services*; Versionone's Project Management tools are used with TestComplete for automated testing environment [26]	**Elastic Infrastructure/Platform, Node-based Availability**; *PaaS, IaaS*; Hudson is a distributed build platform with "master/slave" configuration [27]	**Elastic platform**; *PaaS*; Subversion repository can be accessed by a self-service interface hosted on a shared middleware	**Elastic Platform**; *PaaS*; Bugzilla s hosted on a middleware offered by providers[29]
	Processing Offering	**Hypervisor**; *Virtualization*; JIRA is deployed on virtualized hardware	**Hypervisor**; *Virtualization*; VersionOne can be deployed on virtualized hardware	**Hypervisor**; *Virtualization*; Hudson is deployed on virtualized hardware	**Hypervisor**; *Virtualization*; Subversion is deployed on virtualized hardware	**Hypervisor**; *Virtualization*; Bugzilla is deployed on virtualized hardware
	Storage Offering	**Block**; *Virtualization*; Elastic Bamboo can access centralized block storage thru an API integrated into an operating system running on virtual server[25]	**Block storage**; *Virtualization*; VersionOne can access centralized block storage thru an API integrated into an operating system running on virtual server[26]	**Block, Blob storage**; *Virtualization*; Azure Blob service used as a repository of build artifacts created by a Hudson	**Hypervisor**; *Virtualization*; Subversion can access centralized block storage thru an API integrated into an operating system running on virtual server	**Hypervisor**; *Virtualization*; Bugzilla can access centralized block storage thru an API integrated into an operating system running on virtual server
	Communication Offering	**Message-Oriented Middleware**; *Message Queuing*; JIRA Mail Queue[25]	**Message-Oriented Middleware**; *Message Queuing*; VersionOne's Defect Work Queues[26]	**Message-Oriented Middleware, Virtual networking**;*Message Queuing,Hypervisor*;Hudson's Execution System Queuing component	**Message-Oriented Middleware**;*Message Queuing*;Subversion's Repository layer[28]	**Message-Oriented Middleware**;*Message Queuing*; Bugzilla's Mail Transfer Agent[29]
Cloud Management	Management Components	**Provider Adapter, Managed Configuration, Elastic manager**;*RPC, API*; JIRA Connect Framework[25], JIRA Advanced configuration	**Managed Configuration**;*RPC, API*; VersionOne segregation and appl. configuration	**Elastic load balancer, watchdog**;*Elastic platform*; Hudson execution system's Load Balancer component)	**Managed Configuration**;*RPC, API*; configuration file is used to configure how/when builds are done	**Managed Configuration**;*RPC, API*; Bugzilla can use configuration file for tracking and correcting errors
	Management Processes	**Elastic management process**;*Elasticity Manager*; JIRA Elastic Bamboo, and Time Tracking feature[25]	**Elastic management process**;*Elasticity Manager*; VersionOne's OnDemand security platform[26]	**Update Transition process**;*Message Queuing*; continuous integration of codes by Hudson's CI server[27]	**Update Transition process**;*Message Queuing*; continuous updates of production versions of the appl. by Subversion[28]	**Resiliency management process**;*Elasticity platform*; Bugzilla Bug monitoring/reporting feature[29]
Composite Application	Decomposition Style	**3-tier**;*stateless, processing and data access components*; JIRA is web-based application[25]	**3-tier**;*stateless, processing and data access components*; VersionOne is a web application[26]	**3-tier, Content Dist. Network**;*user interface, processing, data access components, replica distr.*; Hudson is an extensible web application, code file replicated on multiple clouds[27]	**3-tier**;*stateless, processing and data access components*; Subversion is a web-based application [28]	**3-tier**;*stateless, processing and data access components*; Bugzilla is a web application[29]
	Hybrid Cloud Application	**Hybrid processing**; *processing component*; JIRA Agile used to track daily progress work[25]	**Hybrid Development Environment**;*processing component*; VersionOne's OpenAgile Integration[26]	**Hybrid Data, Hybrid Development Environment**; *data access component*;Separate environment for code verification and testing	**Hybrid Data, Hybrid Backup**; *data access component,stateless*;Code files extracted for external storage	**Hybrid Processing**; *processing component*; DB resides in data center, processing done in elastic cloud

support for the implementation of most deployment patterns is practically achieved through API integration [32]. For example, JIRA's Elastic Bamboo support for Blob storage on Windows Azure is through an API [25]. JIRA has a plugin for integrating with Hudson, Subversion and Bugzilla [25] and vice versa. The technologies used to support software processes of GSD tools are highlighted, unlike others which focus mostly on the design of cloud-native applications [6]. Web services (via REST) and messaging (via message queues) are the preferred technologies used by cloud deployment patterns (e.g., stateless pattern) to interconnect GSD tools and other components. REST style is favoured by public cloud platforms. For example, JIRA's support for SOAP and XML-RPC is depreciated in favour of REST [25]. This trend is also reported in [15][32].

D. Patterns for Cloud-application Versus Cloud-environment

Our taxonomy, can be used to guide an architect in focusing on a particular architectural deployment component of interest - that is, either cloud-hosted application or cloud-hosted environment. Other taxonomies [7][15] are concerned with the design of cloud-native applications. Assuming an architect is either interested in providing the right cloud resources, or mapping the business requirement to cloud properties that cannot be changed (e.g., location and ownership of the cloud infrastructure), then Taxonomy A would be more relevant. However, if interest is in mitigating certain cloud properties that can be compensated at an application level (e.g., improving the availability of the cloud-hosted GSD tool), then Taxonomy B should be considered. Fehling et al. describes other cloud properties that are either unchangeable or compensatable [6].

VI. RECOMMENDATIONS

Based on the experience gathered from positioning the GSD tools on the taxonomy, we present a set of recommendations in the form of selection criteria in Table 5 to guide an architect in choosing applicable deployment patterns for deploying any GSD tool. To further assist the architect in making a good choice, we also recommend that the architect should obtain information concerning- (i) the business requirements of the organization; and (ii) the architectural structure, installation

 ISBN: 978-1-61208-388-9

TABLE V. RECOMMENDATIONS FOR SELECTING APPLICABLE DEPLOYMENT PATTERNS FOR CLOUD DEPLOYMENT OF GSD TOOLS.

Category	**Sub-Category**	***Selection Criteria***	***Applicable Patterns***
Application Process	Project Processes	Elasticity of the cloud environment is not required	Static workload
	Implementation Processes	Expects continuous growth or decline in workload over time	Continuously changing workload
	Support Processes	Resources required is nearly constant;continuous decline in workload	Static workload, Continuously changing workload
Core Cloud Properties	Rapid Elasticity	Explicit requirement for adding or removing cloud resources	Elastic platform, Elastic Infrastructure
	Resource Pooling	Sharing of resources on specific cloud stack level-IaaS, PaaS, SaaS	Hypervisor, Standby Pooling Process
	Measured Service	Prevent monopolization of resources	Elastic Infrastructure, Platform, Throttling/Service Metering[7]
Cloud Service Model	Software Resources	No requirement to deploy and configure GSD tool	Software as a Service
	Platform Resources	Requirement to develop and deploy GSD tool and/or components	Platform as a Service
	Infrastructure as a Service	Requires control of infrastructure resources (e.g., storage, memory) to accommodate configuration requirements of the GSD tool	Infrastructure as a Service
Cloud Deployment Model	Private Usage	Combined assurance of privacy, security and trust	Private cloud
	Community Usage	Exclusive access by a community of trusted collaborative users	Community cloud
	Public Usage	Accessible to a large group of users/developers	Public cloud
	Hybrid Usage	Integration of different clouds and static data centres to form a homogenous deployment environment	Hybrid cloud
Application Architecture	Application Components	Maintains no internal state information	User Interface component, Stateless pattern
	Multitenancy	Many different users access and share the same resources	Shared component
	Integration	Integrate GSD tool with different components residing in multiple clouds	Integration provider, Restricted Data Access component
Cloud Offering	Cloud environment	Requires a cloud environment configured to suit PaaS or IaaS offering	Elastic platform, elastic infrastructure
	Processing Offering	Requires functionality to execute workload on the cloud	Hypervisor
	Storage Offering	Requires storage of data in cloud	Block storage, relational database
	Communication Offering	(1) Require exchange of messages internally between appl. components; (2) Require communication with external components	(1) Message-oriented middleware; (2) Virtual Networking
Cloud Management	Management Components	(1) Pattern supports Asynchronous access; (2) State information is kept externally in a central storage	(1) Provider Adapter; Elastic manager; Managed Configuration
	Management Processes	(1)Application component requires continuous update; (2) Automatic detection and correction of errors	(1) Update Transition process; (2) Resiliency management process
Composite Application	Decomposition Style	Replication or decomposition of application functionality/components	(1) 3-tier; (2) Content Distribution Network
	Hybrid Cloud Application	Require the distribution of functionality and/or components of the GSD tool among different clouds	(1) Hybrid processing; (2) Hybrid Data; (3) Hybrid Backup; (4) Hybrid Development Environment

and configuration requirements of the GSD tool (using a process such as IDAPO [5]). Based on this information, a suitable level of cloud stack that will accommodate all the configuration requirements of the GSD tool can be selected. The architect has more flexibility to implement or support the implementation of a deployment pattern when there is greater control of the cloud stack. For example, to implement the hybrid data pattern [6] during cloud deployment of Hudson, the architect would require control of the infrastructure level of the cloud stack to allow for provisioning (and de-provisioning) of resources (e.g., storage, memory, CPU).

VII. LIMITATIONS OF THE STUDY

The GSD tools included in the dataset were stable and mature, and used by all the companies involved in the empirical study. This reduces external threats to the study that may come from the proliferation of GDS tools deployed in the cloud. The study should not be generalized to small and medium size projects. Large projects are usually executed with stable and reliable GSD tools. For small projects (with few developers and short duration), high performance and low cost may be the main consideration in tool selection. The small number of GSD tools in the selected dataset is appropriate because we are not carrying out a feature-analysis based study of GSD tools, but only using it to apply against our proposed taxonomy. Future research can be done to re-evaluate how new GSD tools can be positioned within the taxonomy.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have created and used a taxonomy of deployment patterns for cloud-hosted applications to contribute to the literature on cloud deployment of Global Software Engineering tools.

Eight categories that form the taxonomy have been described: Application process, Cloud properties, Service model, Deployment model, Application architecture, Cloud offerings, Cloud management, and Composite applications. *Application process* contains patterns that handles the workload imposed on the cloud infrastructure by the ISO/IEC 12207 software processes. *Cloud properties* contains patterns for mitigating the core cloud computing properties of the tools. Patterns in *Service model* and *Deployment model* reflects the NIST cloud definition of service models and deployment models, respectively. *Application architectures* contains patterns that supports the architectural components of a cloud-application. Patterns in *Cloud offerings* reflects the main offerings that can be provided to users on the cloud infrastructure. Cloud management contains patterns used to manage both the components and processes of software tools. *Composite cloud* contains patterns that can be formed by combining other patterns or can be decomposed into separate components.

These categories were further partitioned into 24 subcategories, which were mapped to the components of an (architectural) deployment structure. This mapping reveals two components classes: cloud-hosted environment and cloud-hosted application. Cloud-hosted environment and cloud-hosted application classes captures patterns that can be used to address deployment challenges at the infrastructure level and application level, respectively.

By positioning a selected set of software tools, JIRA, VersionOne, Hudson, Subversion and Bugzilla, on the taxonomy, we were able to identify applicable deployment patterns together with the supporting technologies for deploying cloud-hosted GSD tools. We observed that most deployment patterns are related and can be implemented by combining with others ones, for example, in hybrid deployment scenarios to integrate data residing in multiple clouds. We have also provided recommendations in a tabular form which shows the selection criteria to guide an architect in choosing applicable deployment patterns. Examples of deployment patterns derived

 ISBN: 978-1-61208-388-9

from applying these selection criteria have been presented.

We plan to carry out several Case Studies involving the deployment of cloud-hosted GSD tools to compare how well different deployment patterns perform under different deployment conditions with respect to specific software development processes (e.g., continuous integration with Hudson). In the future, we will evaluate performance and reliability (through simulation) in multi-user collaborations involving cloud-hosted GSD tools for different deployment patterns.

ACKNOWLEDGMENT

This research was supported by the Tertiary Education Trust Fund (TETFUND), Nigeria and IDEAS Research Institute, Robert Gordon University, UK.

REFERENCES

[1] R. Buyya, J. Broberg, and A. Goscinski, Cloud Computing: Principles and Paradigms. John Wiley & Sons, Inc., 2011.

[2] M. A. Chauhan and M. A. Babar, "Cloud infrastructure for providing tools as a service: quality attributes and potential solutions," in Proceedings of the WICSA/ECSA 2012 Companion Volume. ACM, 2012, pp. 5–13.

[3] S. Junuzovic and P. Dewan, "Response times in n-user replicated, centralized, and proximity-based hybrid collaboration architectures," in Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work. ACM, 2006, pp. 129–138.

[4] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 3/E. Pearson Education India, 2013.

[5] K.-J. Stol, P. Avgeriou, and M. A. Babar, "Design and evaluation of a process for identifying architecture patterns in open source software," in Software Architecture. Springer, 2011, pp. 147–163.

[6] C. Fehling, F. Leymann, R. Retter, W. Schupeck, and P. Arbitter, Cloud Computing Patterns. Springer, 2014.

[7] A. Homer, J. Sharp, L. Brader, M. Narumoto, and T. Swanson, Cloud Design Patterns, R. Corbisier, Ed. Microsoft, 2014.

[8] L. Badger, T. Grance, R. Patt-Corner, and J. Voas, "Cloud computing synopsis and recommendations," NIST special publication, vol. 800, 2012, p. 146.

[9] S. Hansma, "Go fast and don't break things: Ensuring quality in the cloud." in Workshop on High Performance Transaction Systems(HPTS 2011), Asilomar, CA, October 2011. Summarized in Conference Reports column of USENIX; login 37(1), February 2012., 2012.

[10] J. Bass, "How product owner teams scale agile methods to large distributed enterprises," Empirical Software Engineering, 2014, pp. 1–33.

[11] F. Lanubile, "Collaboration in distributed software development," in Software Engineering. Springer, 2009, pp. 174–193.

[12] J. D. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," Software Engineering, IEEE Transactions on, vol. 29, no. 6, 2003, pp. 481–494.

[13] J. Portillo-Rodriguez, A. Vizcaino, C. Ebert, and M. Piattini, "Tools to support global software development processes: a survey," in Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on. IEEE, 2010, pp. 13–22.

[14] J. Vlissides, R. Helm, R. Johnson, and E. Gamma, "Design patterns: Elements of reusable object-oriented software," Reading: Addison-Wesley, vol. 49, 1995.

[15] B. Wilder, Cloud Architecture Patterns, 1st ed., R. Roumeliotis, Ed. 1005 Gravenstein Highway North, Sebastopol, CA 95472.: O'Reilly Media, Inc., 2012.

[16] J. Buckley, T. Mens, M. Zenger, A. Rashid, and G. Kniesel, "Towards a taxonomy of software change," Journal of Software Maintenance and Evolution: Research and Practice, vol. 17, no. 5, 2005, pp. 309–332.

[17] D. Smite, C. Wohlin, Z. Galvina, and R. Prikladnicki, "An empirically based terminology and taxonomy for global software engineering," Empirical Software Engineering, 2012, pp. 1–49.

[18] C. Moyer, Building Applications for the Cloud: Concepts, Patterns and Projects. Pearson Education, Inc, Rights and Contracts Department, 501 Boylston Street, Suite 900, Boston, MA 02116, USA: Addison-Wesley Publishing Company, 2012.

[19] Z. Mahmood, Ed., Cloud Computing: Methods and Practical Approaches. Springer-Verlag London, 2013.

[20] N. Sawant and H. Shah, Big Data Application Architecture - A problem Solution Approach. Apress, 2013.

[21] S. Strauch, U. Breitenbuecher, O. Kopp, F. Leymann, and T. Unger, "Cloud data patterns for confidentiality," in Proceedings of the 2nd International Conference on Cloud Computing and Service Science, CLOSER 2012, 18-21 April 2012, Porto, Portugal. SciTePress, 2012, pp. 387–394.

[22] J. Varia, "Migrating your existing applications to the cloud a phase-driven approach to cloud migration." Amazon Web Services (AWS), [Online: retrieved in October, 2014 from http://aws.amazon.com/whitepapers/].

[23] L. Lilien, "A taxonomy of specialized ad hoc networks and systems for emergency applications," in Mobile and Ubiquitous Systems: Networking & Services, 2007. MobiQuitous 2007. Fourth Annual International Conference on. IEEE, 2007, pp. 1–8.

[24] P. Mell and T. Grance, "The nist definition of cloud computing," NIST special publication, vol. 800, no. 145, 2011, p. 7.

[25] Atlassian.com. Atlassian documentation for jira 6.1. Atlassian, Inc. [Online: retrieved in November, 2015 from https://www.atlassian.com/software/jira/].

[26] VersionOne, "Versionone-agile project management and scrum," [Online: retrieved in November, 2014 from www.versionone.com/].

[27] M. Moser and T. O'Brien, "The hudson book." Oracle, Inc., USA, [Online: retrieved in November, 2014 from http://www.eclipse.org/hudson/the-hudson-book/book-hudson.pdf].

[28] B. Collins-Sussman, B. Fitzpatrick, and M. Pilato, Version control with subversion. O'Reilly, 2004.

[29] Bugzilla.org. The bugzilla guide. [Online: retrieved in October, 2014 from http://www.bugzilla.org/docs/].

[30] CollabNet. Subversionedge for the enterprise. CollabNet, Inc. [Online: retrieved in October, 2014 from http://www.collab.net/products/subversion].

[31] I. Sommerville, Software Engineering. Pearson Education, Inc. and Addison-Wesley, 2011.

[32] J. Musser. Enterprise-class api patterns for cloud and mobile. CITO Research. (2012)

 ISBN: 978-1-61208-388-9

Residual Traffic Based Task Scheduling in Hadoop

Daichi Tanaka
University of Tsukuba
Graduate School of Library, Information and Media Studies
Tsukuba, Japan
e-mail: s1421593@u.tsukuba.ac.jp

Masatoshi Kawarasaki
University of Tsukuba
Faculty of Library, Information and Media Science
Tsukuba, Japan
e-mail: mkawa@slis.tsukuba.ac.jp

Abstract— **In Hadoop job processing, it is reported that a large amount of data transfer significantly influences job performance. In this paper, we clarify that the cause of performance deterioration in the CPU (Central Processing Unit) heterogeneous environment is the delay of copy phase due to the heavy load in the inter rack links of the cluster network. Thus, we propose a new scheduling method -Residual Traffic Based Task Scheduling- that estimates the amount of inter rack data transfer in the copy phase and regulates task assignment accordingly. We evaluate the scheduling method by using ns-3 (network simulator-3) and show that it can improve Hadoop job performance significantly.**

Keywords- distributed computing; Hadoop; MapReduce; job performance; network simulation.

I. INTRODUCTION

Owing to the rapid reduction in hard disk drive (HDD) cost and the rapid expansion in the variety of services, the amount of data volume in the world is ever increasing. Along with it, there are ongoing efforts worldwide to extract useful information from a large amount of data (big data) [1]. Under these circumstances, MapReduce [2] was developed as a distributed processing framework for big data. In MapReduce, data processing is performed in two stages, namely map stage and reduce stage. By performing parallel distributed processing in each stage, big data can be processed at high speed. Hadoop [3] is an open source framework of MapReduce that implements the functionalities to deal with problems such as fault tolerance, load distribution and consistency. As a result, Hadoop removed many difficulties of conventional distributed processing, thus making many enterprises such as Facebook, Yahoo and New York Times to use it for site management or log analysis.

Since Hadoop has become popular, research on improving Hadoop performance is being actively carried out in pursuit of a more efficient scheduling scheme [4]-[9]. However, these studies have been focusing on scheduling computation and storage resources, while mostly ignoring network resources. In [10], it is reported that data transfer time may account for more than 50% of total job execution time. In this paper, we explore how data transfer affects Hadoop job performance. In particular, we analyze how network congestion deteriorates overall job performance and, based on this analysis, we propose enhancements to Hadoop scheduler by introducing the concept of residual traffic over the network.

For the study of Hadoop performance, we need to build a large-scale experimental environment. On the other hand, the scale of the production environment of Hadoop is very large, having several hundred to several thousand nodes [1]. Since building the actual size of the cluster is not realistic, many studies described above use cloud services such as Amazon Elastic Compute Cloud (Amazon EC2) [11] to construct the experimental environment. In the cloud service, although the node performance (e.g., CPU and HDD) and the minimum bandwidth of inter-node links are guaranteed, the network topology is opaque. Furthermore, resource usage may be affected by other users of the cloud. Accordingly, we have developed Hadoop cluster simulator using ns-3 (network simulator-3) [12] [13] so that we can set the size and the topology of a network freely.

Using this Hadoop cluster simulator, we perform Hadoop simulation using sort benchmark. Through this experiment, we clarify that the cause of performance deterioration in the CPU heterogeneous environment is the delay of copy phase due to the heavy load in the inter rack links of the cluster network. Based on this analysis, we propose a new scheduling method -Residual Traffic Based Task Scheduling- that estimates the amount of inter rack data transfer in the copy phase and regulates task assignment accordingly. We evaluate the proposed scheduling method and show that it can improve Hadoop job performance significantly.

The rest of this paper is organized as follows. Section II provides an overview of Hadoop architecture and mechanism. Section III and IV describe the Hadoop cluster simulator and experiment discussing performance issues. Based on this, Section V proposes residual traffic based task scheduling to improve Hadoop performance and Section VI discusses its effectiveness. Section VII concludes this paper.

II. HADOOP OVERVIEW

A. Hadoop Structure

Hadoop consists of MapReduce engine and Hadoop Distributed File System (HDFS) as shown in Figure 1. Hadoop cluster is made of one MasterNode and many SlaveNodes. The MasterNode and the SlaveNode communicate with each other using the HeartBeat mechanism. The MasterNode receives HeartBeat message from a

 ISBN: 978-1-61208-388-9

SlaveNode at a constant frequency. It includes the state information of the SlaveNode.

In MapReduce engine, the MasterNode is called JobTracker and the Slave Node is called TaskTracker. When JobTracker receives a job from a user, it divides it into small tasks and assigns them to TaskTrackers. When a TaskTracker requests a task assignment to JobTracker by sending a HeartBeat message, the JobTracker assigns a task in response to this so that the number of assigned tasks does not exceed the number of taskslots whose value is pre-determined according to the performance of each TaskTracker node.

In HDFS, the MasterNode is called NameNode and the SlaveNode is called DataNode. When NameNode receives data from a user, it splits it into small file blocks (called chunks) having 64MB in size, and distributes them to DataNodes. At this time, NameNode produces replica to improve fault tolerance and reachability.

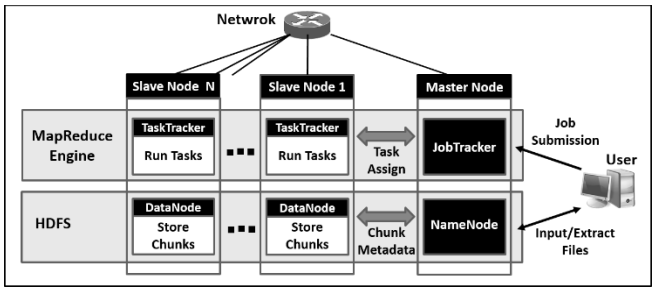


Figure 1. Hadoop Structure

B. MapReduce Procedures

MapReduce is a distributed processing framework for big data that processes data by using map function and reduce function that are defined by a user. Figure 2 shows MapReduce processing flow.

When a user submits a job, the JobTracker receives the input data split of the user data and generates map tasks. One map task is generated for one input data split. After the Job is turned on and the HeartBeat arrives at the JobTracker from a TaskTracker, the JobTracker assigns as many map tasks as the number of free map taskslot of the TaskTracker. In this case, a map task is assigned preferentially to the TaskTracker that holds the input data split to be processed. This map task is called a data local map task. If a map task is not data local, input data split needs to be obtained from other node over a network. After obtaining the input data split, the map function is executed. The result (map output) is stored in HDFS. When the map task is successfully completed, the TaskTracker sends a termination notice to the JobTracker.

When a part of map tasks of a given job (5% by default) are completed, JobTracker starts to assign reduce tasks to TaskTrackers. Reduce task collects all the map output having the same key over the network (copy phase) and performs reduce function. The results are stored in HDFS. When the reduce task is successfully completed, the TaskTracker sends a termination notice to JobTracker. JobTracker sends the Job completion notification to the user when it confirmed the end of all of tasks, and the job completes.

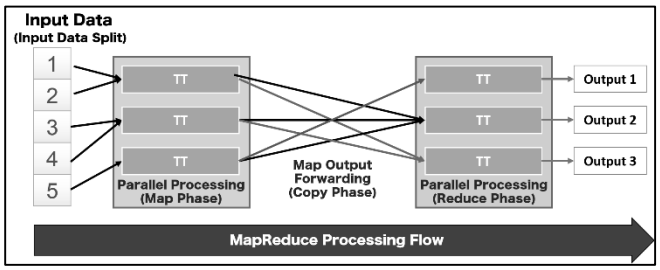


Figure 2. MapReduce Procedures

C. Hadoop Network I/O

During the MapReduce procedures, data transfer occurs in the following occasions.

1) Transfer of input data split: When assigned map task is not data local, input data split is obtained from other node. In this case, transmission of the size of input data split (by default 64MB) is generated.

2) Transfer of map output data: Reducer node receives all the relevant map outputs from mapper nodes. In this case, transmission of the total amount of map outputs is generated. We define this as resilient traffic in the later section of this paper.

3) Transfer of reduce output replica:

As reduce task output data (i.e., Job output data) is stored in the HDFS, transmission of copies corresponding to the number of replicas is generated. By default, three replicas are generated. One of the replicas is stored on the disk of the reducer node and the other two are stored in other nodes

III. HADOOP CLUSTER SIMULATOR

A. Design Principle

The objective of Hadoop cluster simulator is to provide experimental environment of Hadoop cluster network whose size and topology can be set freely. It simulates Hadoop job scheduling and task scheduling as defined in Hadoop version 1.1.2 and the three kinds of data transfer as described in Section II-C. We designed the simulator to keep track of cooperation with Hadoop scheduler behavior and network behavior.

B. Input Parameters

Although there are many parameters that can be set to actual Hadoop, the parameters that affect the performance of Hadoop are limited. The parameters used in our simulation are shown in Table I.

TABLE I. SIMULATION PARAMETERS

Category	*Component*
Job parameter	Scheduling method (FIFO, Fair), Job type (sort, search, index, etc.)
Configuration parameter	Number of taskslots, Chunk size, Number of replicas
Cluster parameter	Network topology, Number of nodes

Job parameters include Job Scheduling methods such as FIFO (First In First Out) Scheduling and Fair Scheduling, and Job types such as *sort*, *search* and *index*. They affect data transfer in the data generation timing and the amount of data. Specific values are shown in Table II and explained in section C.

The Configuration Parameters are the parameters that are set in the actual Hadoop configuration file. What affects data transfer is the block size when non-data local map task obtains input data split from other node and the number of replicas when the job output is copied to HDFS. The maximum number of task slots that is pre-allocated to each node affects the data transfer as well. If the number of slots is large, network traffic might occur in bursts and if the number of slots is not uniform among TaskTrackers, imbalance might occur in link bandwidth utilization.

The Cluster Parameters are the parameters that determine a network configuration of the cluster, such as the network topology and the number of nodes.

C. Modelling of Map Task and Reduce Task Processing

Key parameters that determine Hadoop performance are task processing time and output data size. We assumed the following:

1) Task Processing Time: Task processing time is determined only by the data size, not by the data content. Specific values are shown in Table II. These values were determined by reference to [12] and the actual measurement value in our experimental Hadoop cluster made of real machines [14].

2) Disk I/O Time: The time needed for disk I/O is negligibly small.

Output data size is calculated by "Filter" value. Filter means the ratio of output data size to input data size.

TABLE II. SPECIFIC PARAMETER VALUES

Parameters	Job Type		
	sort	*search*	*index*
MapFilter (%)	100	0-0.1	2-50
ReduceFilter (%)	100	100	2-50
MapProcessSpeed (sec/MB)	0.03	0.16	0.016
Reduce ProcessSpeed (sec/MB) (Sort phase)	0.016	0.016	0.016
Reduce ProcessSpeed (sec/MB) (Reduce phase)	0	0	0

Reduce task is divided into copy, sort and reduce phases. Reduce tasks can start when only some map tasks complete (by default 5%), which allows reduce tasks to copy map outputs earlier as they become available and hence mitigates network congestion. However, no reduce task can step into the sort phase until all map tasks complete. This is because each reduce task must finish copying outputs from all the map tasks to prepare the input for the sort phase.

The amount of data transfer in the copy phase is determined by "(map output data size) / (the number of reduce tasks)". As the reduce output (i.e., job output) will be copied to other nodes by the HDFS, the data transfer amount generated at replication depends on the size of the reduce task output.

D. Validation of Developed Simulator

To validate the accuracy of the developed simulator, we performed comparative experiments with our experimental Hadoop cluster made of Amazon EC2[14]. The comparison scenario is shown in Table III. Validation was focused on scheduler behavior and network I/O of TaskTrackers.

TABLE III. COMPARISON SCENARIO

Parameter	*Component*
# Nodes	JobTracker : 1, TaskTracker :20
# Task slots	map slot = 4, reduce slot = 4
Clunk Size	64 MB (by default)
Job Type	sort
# Job	1
Job Size	10 GB

1) Scheduler Behavior: We validated the accuracy of task scheduling for the followings:

a) Priority handling of Data Local Map Task: Basically, map tasks are processed in the order of task ID. However, if a particular map task is not data local, the assignment of that map task is skipped.

b) Concurrent Processing of Map Tasks: If the number of map tasks of a given job is greater than the total number of map task slots within a cluster, all the map tasks cannot be assigned in one cycle. In this case, map task assignment is achieved in several cycles.

c) Start of Reduce Task: Reduce task can starts to be assigned when a certain amount of map tasks are completed.

d) Phased Processing of Reduce Task: Sort and reduce phases cannot be started until copy phase is completed.

By comparing the task progress gantt charts of Hadoop EC2 cluster and the developed simulator as well as examining simulation logs, we confirmed that these behaviors are accurately simulated.

2) Network I/O of TaskTrackers: We examined whether three kinds of data transfer as described in Section II-C occur at suitable timing. In our experiment, we measured the throughput at the network interface of a TaskTracker that is assigned one non data local map task and some reduce tasks. The result is shown in Figure 3. From Figure 3 and experimental log, we confirmed the following:

i. Transfer of input data split occurs at the timing of non-data local map task assignment.
ii. Transfer of map output data occurs at the timing of reduce task assignment.
iii. Transfer of reduce output replica occurs at the end of reduce task.

From the above, we have confirmed that this simulator correctly simulates Hadoop cluster behavior.

 ISBN: 978-1-61208-388-9

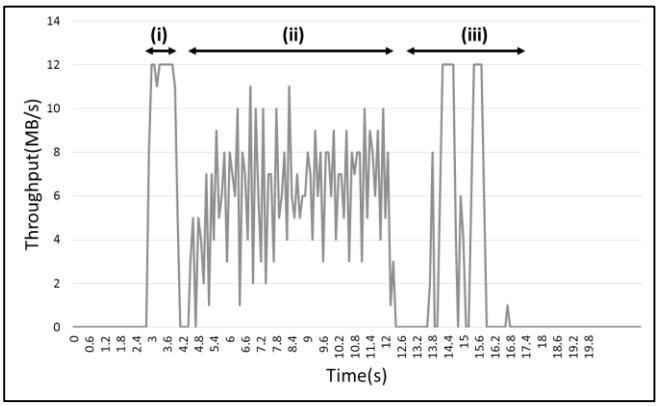


Figure 3. Network Input Throughput at a TaskTracker

IV. EXPERIMENT OF HADOOP PERFORMANCE

By using the developed simulator, we carried out Hadoop performance experiment.

A. Experiment Setting

The network model of experimental Hadoop cluster is shown in Figure 4. It has a tree structure (double-star type) that is common in many datacenters. It consists of 4 racks each of which includes 10 worker nodes. In total, there are 40 TaskTrackers (TTs) and one JobTracker (JT).

Each worker node is connected to Top of Rack switch (ToR-SW) by 1Gbps link. We call these links ToR-TT link in this paper. Each ToR switch is connected to Aggregate switch (AGG-SW) by 2.5Gbps link. As for the links between AGG-SW and Rack i, we call the uplink UP(i) and the downlink DOWN(i). Regarding CPU performance, we assumed heterogeneous environment. The CPU processing speed is faster in Rack 0, normal in Racks 1 and 2, and slower in Rack 3. The specific values are shown in Table IV. Conforming to CPU processing speed, the number of pre-allocated task slots in worker node is 8 for Rack 0, 4 for Rack 1 and 2, and 2 for Rack 3. We assumed homogeneous environment within each rack.

Job characteristics are summarized in Table V. In our experiment, a *sort* job having 5Gbytes was submitted every 5 seconds to the cluster. A total of 10 jobs were submitted in one experiment. We used sort benchmark that generates a large amount of output data. This is because we focused on the analysis of data transfer in Hadoop performance.

As for Job Scheduling, we used Fair Scheduler [5]. In Fair Scheduler, jobs are grouped into pools. When a slot needs to be assigned, the job that has been most unfair is scheduled. This time, we implemented to assign one pool for each job. Accordingly, all the running jobs are scheduled to obtain task slots equally.

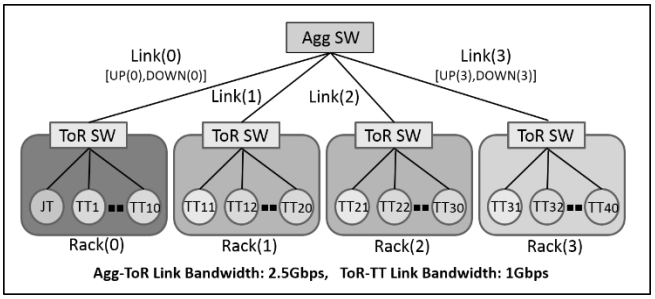


Figure 4. Experiment Cluster Architecture

TABLE IV. NODE PARAMETERS

Rack	*CPU performance*	*# Map/Reduce taskslot*	*Processing Speed (Relative Value)*
R0	Faster	8	2.0
R1, R2	Normal	4	1.0
R3	Slower	2	0.5

TABLE V. SIMULATION PARAMETERS

Category	*Component*	*Value*
Job Parameter	Scheduling Method	Fair Scheduling
	Job Type	Sort
	Job Size	5GB
Configuration Parameter	#Taskslots per TT	2~8
	Chunk Size	64MB(default)
	#Replication	3
Cluster Parameter	#TaskTracker (TT)	40

B. Experiment Results

Figure 5 shows the job execution time. We can see that job execution time becomes significantly long after Job5. Figure 6 shows the phase duration time of map task and reduce task (divided into copy phase and sort phase) of Job 0 and Jobs 7, 8 and 9. Regarding map task, we distinguish node local map task, rack local map task and remote map task.

For a given job, the meaning of phase duration time is as follows: Map Phase is the period from when the Job is turned on until the end of the final map task. Copy phase is the period from the start of the copy transmission of the first reduce task until the end of copy transmission of the last reduce task. Sort+Reduce Phase is the period from the beginning of reduce task that entered into the sort phase first until the end of the last reduce phase (i.e., the end of the job).

From Figure 6, we can see that Map Phase is very short and completed immediately in each job. On the contrary, copy phase is very long for Job 7, 8, 9 whose job execution time is long. Similarly, Job 7, 8, 9 take a very long time since Map Phase is finished to Sort+Reduce Phase begins, compared to Job 0.

 ISBN: 978-1-61208-388-9

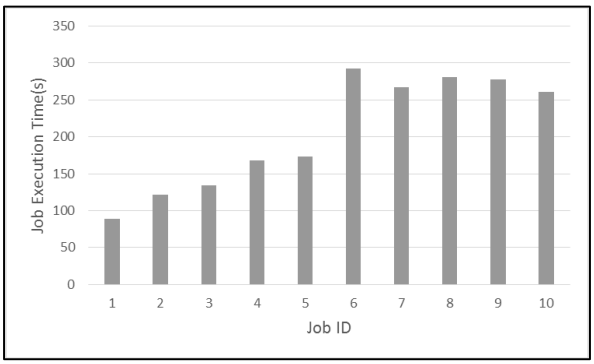


Figure 5. Job Execution Time per Jobs

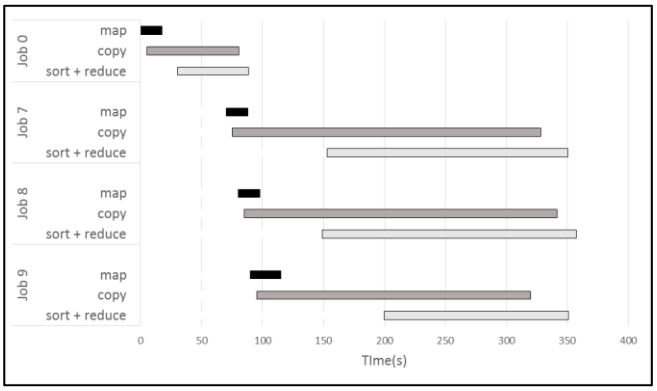


Figure 6. Phase Duration Time per Jobs（Job 0, Job 7, Job 8, Job9）

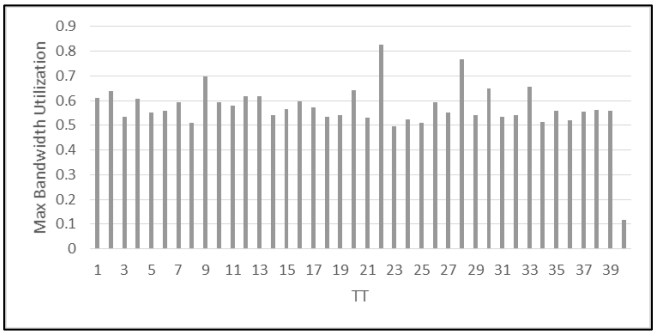


Figure 7. Max Bandwidth Utilization (TT-ToR)

Next, in order to examine the factors that take time to copy phase, we measured the maximum utilization of the each TT-ToR downlink and the bandwidth utilization of UP(i) and DOWN(i) at AGG-ToR links. Each is shown in Figure 7, 8 and 9. From Figure 7, the maximum utilization of the TT-ToR links is 0.6 or at most 0.8. It can be seen that these links are underutilized through the Job execution time. By contrast, from Figure 8 and 9, UP(i) and DOWN(i) links are highly utilized. Especially, UP (0) link maintains 100% of utilization for a long period of time.

Figure 10 shows the number of processing Tasks per rack. Here, the Remote MapTask is a map task that is neither node local nor rack local. Since we are using slot-dependent scheduling scheme (a task is assigned immediately if a taskslot is opened), the number of processing tasks of each

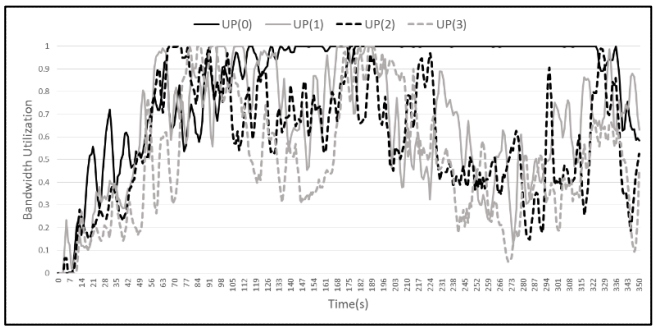


Figure 8. Uplink Bandwidth Utilization (UP)

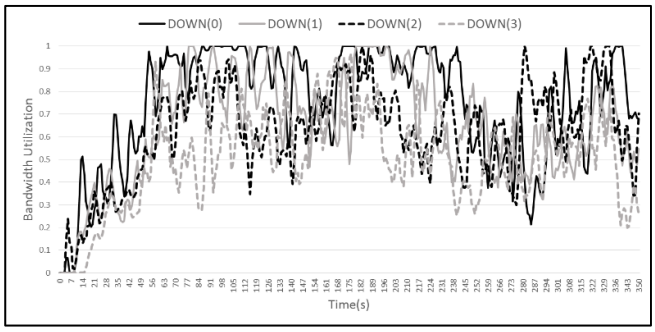


Figure 9. Downlink Bandwidth Utilization (DOWN)

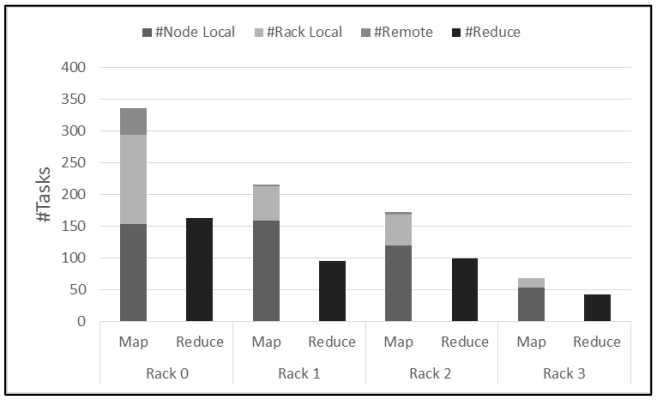


Figure 10. Number of Tasks per Rack

rack is proportional to the total number of slots allocated to the rack. From Figure 10, we can see that Rack 0 has processed a larger number of tasks compared to other racks. In particular, the number of map processing is remarkable.
Performance Deterioration Mechanisms

From the above experimental results, the performance deterioration is inferred to occur in the following manner. First, in CPU heterogeneous environment, there always is a rack having faster nodes than others. In map task, as most of the retention time of taskslot is CPU processing time, the faster rack processes a large amount of map tasks at high speed. It can be confirmed from Figure 10 that the map task processing is concentrated in Rack 0. This phenomenon is called "Task Stealing" [15]. If map tasks concentrate on a particular rack, a large volume of map output requests occur from inside and outside of the rack. As a result, the uplink of the faster rack

 ISBN: 978-1-61208-388-9

becomes highly used. This can be confirmed by Figure 8. Bottleneck at UP (0) prolongs the copy phase of each reduce task, thus deteriorating Job performance.

From the above discussion, we conclude that the cause of performance deterioration in the CPU heterogeneous environment is the delay of copy phase due to the heavy load in the inter rack link. Inter rack data transfer is generated in the following three cases:

(1) Input data split transfer caused by Remote MapTask,
(2) Copy phase transfer caused by ReduceTask,
(3) Job output transfer caused by replication.

Among these, Remote MapTassks are very few, as seen from Figure 10. Replication is not under the control of HDFS and out of the scope of Task Scheduling. Thus, we propose a new scheduling method -Residual Traffic Based Task Scheduling- that reflects the inter rack data transfer in the copy phase in task scheduling.

V. RESIDUAL TRAFFIC BASED TASK SCHEDULING

Based on the analysis described above, we propose enhancements to Hadoop task scheduling to improve Hadoop performance. Our proposal makes Hadoop scheduler aware of network congestion and regulates task assignment proactively. In this section, we propose the enhanced task scheduling algorithm.

A. Residual Traffic

To predict the inter rack link congestion status, we define residual traffic of Rack *i*. Before describing the residual traffic, we define the residual transmission amount and the residual reception amount. The residual transmission amount is the total amount of the map output data that was already generated by map tasks but have not being received by relevant reduce tasks. The residual reception amount is the total amount of the map output data that has not been received yet by relevant reduce tasks since their assignment.

Residual Up Traffic is the sum of the residual transmission amount of map output in each rack, and Residual Down Traffic is the sum of residual reception amount of running reduce tasks in each rack. Accordingly, ResidualUpTraffic (i) and ResidualDownTraffic (i) can be calculated as follows:

$$\text{ResidualUpTraffic(i)} = \sum_{j=1}^{number\_of\_map(i)} RemainData(MAP(i,j)) \quad (1)$$

$$\text{ResidualDownTraffic(i)} = \sum_{j=1}^{number\_of\_red(i)} RemainData(RED(i,j)) \quad (2)$$

Here, Map(i,j) is a map task that was assigned to Rack i in j-th order, RemainData (MAP) is the total amount of map output data of a given map task whose transmission is not completed, RED (i, j) is the ReduceTask that was assigned to Rack i in j-th order, and RemainData (RED) is the total amount of map output data that a given reduce task has not received yet. By calculating residual traffic, we can predict the load of inter rack links in the immediate future. If a rack has a large Residual Up Traffic, we can predict that the uplink from the rack is likely to congest. Similarly, if a rack has a large Residual Down Traffic, we can predict that the downlink to the rack is likely to congest.

TABLE VI. NUMBER OF ASSIGNABLE TASKS IN EACH REGULATORY LEVEL

Level	*Green*	*Yellow*	*Red*
#Assignable MapTasks	Available Map Slot	1	0
#Assignable ReduceTasks	1	0	0

B. Scheduling Algorithm

In the residual traffic based scheduling, JobTracker monitors the inter rack link bandwidth utilization, and if the usage has exceeded the threshold level, it adjusts the regulatory level in task assignment of each rack (i.e., UPRegulatoryLevel and DownRegulatoryLevel) referring to the residual transmission amount and/or residual reception amount. We discriminate three stages of regulatory level (Green, Yellow and Red) depending on the combined status of link utilization and residual traffic amount, so that we can change the way of task assignment accordingly. Table VI shows the number of tasks that can be assigned in each of the regulatory levels.

- *Regulatory level (Green):* Normal Task Scheduling. No regulation is applied.
- *Regulatory level (Yellow):* MapTask can be assigned one at most. ReduceTask cannot be assigned at all.
- *Regulatory level (Red):* No Map task or Reduce Task can be assigned.

Regulatory level is updated at every HeartBeat communication. Update algorithm for UPRegulatoryLevel of Rack *i* is shown below. DownRegulatoryLevel is updated similarly.

Algorithm Regulatory Level Update Algorithm

```
WHEN JobTracker receive a HeartBeat from a TaskTracker in Rack i
  IF UPRegulatoryLevel(i) = Green  THEN
    IF UpBandUsage(i) > StartTH
      IF ResidualUpTraffic(i) > Th_yr THEN
        Set UPRegulatoryLevel(i) to Red
      END IF
      IF ResidualUpTraffic(i) < Th_yr THEN
        Set UPRegulatoryLevel(i) to Yellow
      END IF
  END IF
END IF
  IF UPRegulatoryLevel(i) = yellow or red THEN
    IF UpBandUsage(i) < EndTh THEN
      Set UPRegulatoryLevel(i) to Green
    END IF
  END IF
END WHEN
```

Figure 11. Regulatory Level Update Algorithm

In the above, UpBandUsage(i) is the uplink bandwidth utilization of Rack i between ToR Switch and Aggregation Switch, DownBandUseage(i) is the downlink bandwidth

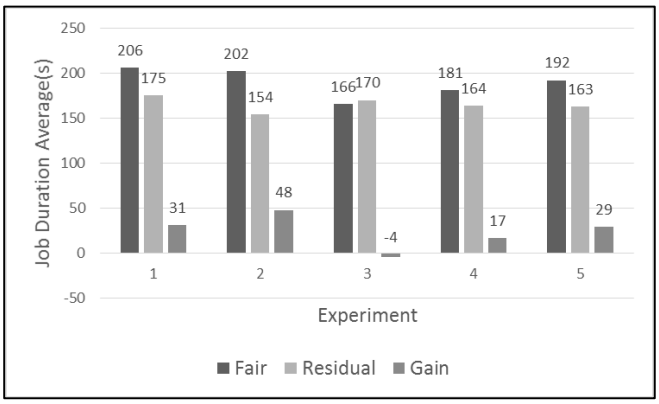


Figure 12. Average Job Execution Time in Each Experiment

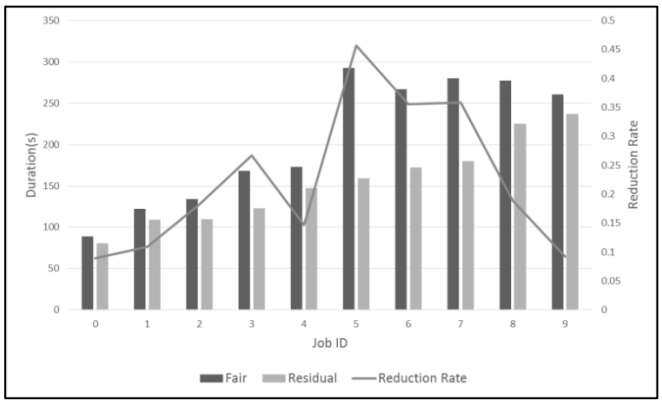


Figure 13. Job Execution Time Per Jobs

TABLE VII. RESIDUAL TRAFFIC LOAD BASED SCHEDULING PARAMETER

Parameter	*Value*
StartTH	0.8
EndTH	0.6
Th_yr	3.0

utilization of Rack i between ToR Switch and Aggregation Switch, StartTH is the bandwidth utilization threshold to start regulation, EndTH is the bandwidth utilization threshold to exit regulation, and Th_yr represents the boundary of residual traffic between strong regulation (Red) and weak regulation (Yellow).

StartTH and EndTH affect the strength and the duration of regulation. If StartTH is low, the regulation is easily invoked. If the difference between StartTH and EndTH is large, the duration of the regulation becomes long, because once the regulation is turned on, it is less likely to off. Th_yr also is a parameter related to the strength of the regulation. If Th_yr is low, regulatory level is likely to be Red and strong regulation is applied. Th_yr is a parameter that needs to be properly adjusted according to the execution environment.

VI. EVALUATION

In this section, we evaluate the Residual Traffic Based Scheduling described in Section V. In the experiment, we used the Hadoop cluster simulator described in Section III. We performed simulation five times by changing only the Scheduling scheme in the same scenario as preliminary experiments. The parameters used in the Residual Scheduling are shown in Table VII. The parameter values in Table VII were determined empirically to maximize the effectiveness of the proposed method in this execution scenario.

A. Experiment Results

1) Job Execursion Time: After performing the experiments five times, we took the average of Job execution time for each Scheduling method. The result is shown in Figure 12. The average reduction in Job execution time was 24.2s and the reduction rate was about 13%. The maximum reduction in Job execution time was 48s and reduction rate was 23%. From

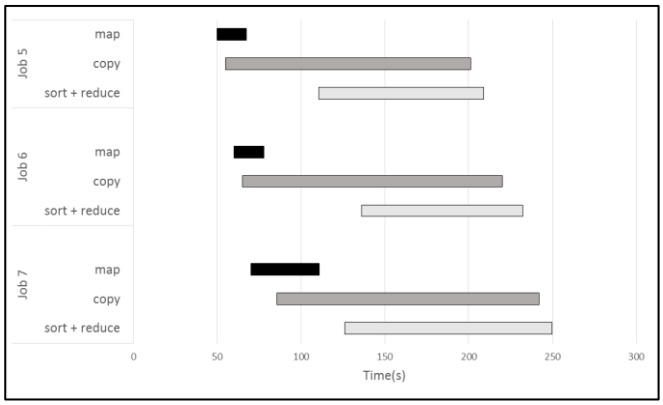


Figure 14. Phase Duration Time per Jobs (Residual Scheduling)

here, we use the results of the third experiment as a result of Residual Scheduling. Figure 13 shows the Job execution time of Fair and Residual Scheduling and the reduction rate. The average, maximum and minimum job execution time was 154s, 236s and 80s, respectively. The maximum reduction rate was 46% and it appeared in Job 5.

2) Phase Duration Time: To explore the cause of job execution time improvement, we analyse the duration of each phase. Figure 14 shows the phase duration time of Job 5, 6, and 7 whose reduction rate were large in Figure 13. In all of these jobs, the time difference between the end time of Map Phase and the start time of Sort + Reduce Phase is small, compared to Fair scheduling that is shown in Figure 6. To examine the factors that shorten this interval, we investigate the state of network link utilization during the time from 50s to 250s which corresponds to the copy phase of Jobs 5, 6 and 7.

3) Inter Rack Link Bandwidth Utilization: To evaluate the effectiveness of Residual Scheduling, we measured the bandwidth utilization of the uplink and downlink between

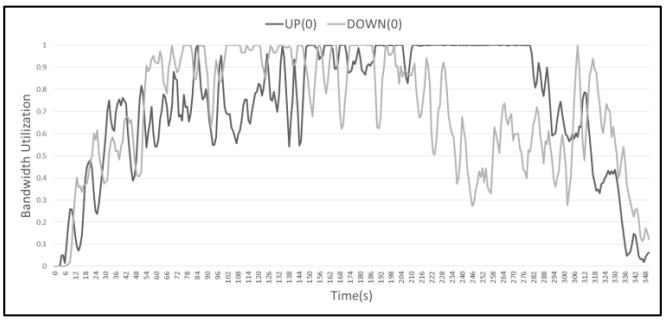


Figure 15. Bandwidth Utilization (Residual Scheduling)

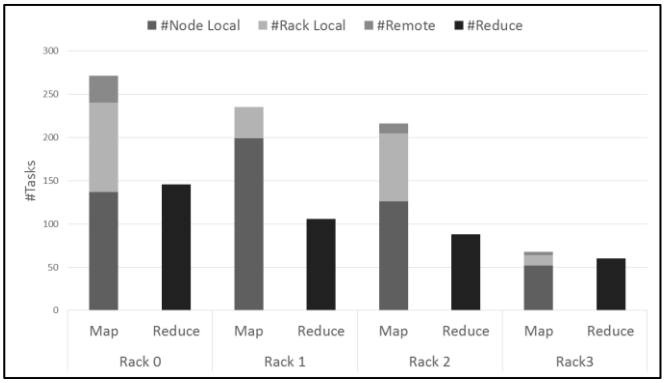


Figure 16. Number of Tasks per Rack

AGG-SW and ToR-SW of Rack 0 which was congested under Fair Scheduling. Figure 15 shows the measurement results. When compared with Figure 8 and 9, bandwidth utilization is improved during 50 ~ 250s. To investigate the reason of this improvement, Figure 16 shows the number of tasks per rack. When compared with Figure 10 which is the result of Fair Scheduling, the difference between Rack 0 (faster CPU*)* and Rack 1 & 2 (normal speed CPU) is shrinking.

4) Residual Traffic: Figures 17 and 18 show the Residual Up Traffic of each rack under Fair Scheduring and Residual Scheduling, respectively. The difference in resudial traffic between racks is equalized under Residual Scheduling.

B. Effectiveness of Residual Traffic Based Scheduling

From Figures 17 and 18, we can see that the residual amount of transmission is regulated so as not to concentrate on Rack 0. This is also supported from the fact that the number of MapTasks assigned to Rack 0 is not concentrated as shown in Figure 16. However, even under the Residual Scheduling, the link bandwidth utilization UP (0) becomes 100% at the later stage of simulation as shown in Figure 18. This is because of the replication of job output performed by the HDFS and cannot be regurated by task scheduling. As we used Sort job, twice the amount of job size of data transfer occurs when the number of Replica is three, thus made UP(0) bandwidth utilization 100%.

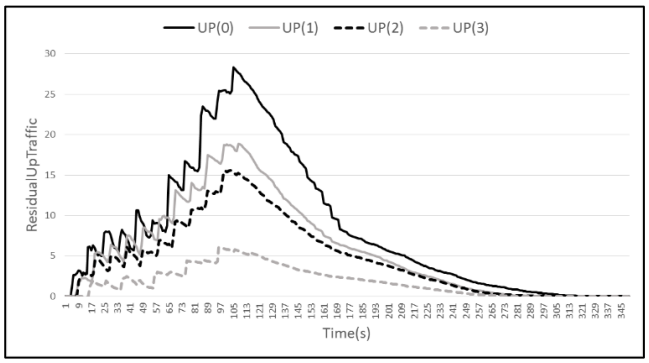


Figure 17. ResidualUpTraffic (Fair)

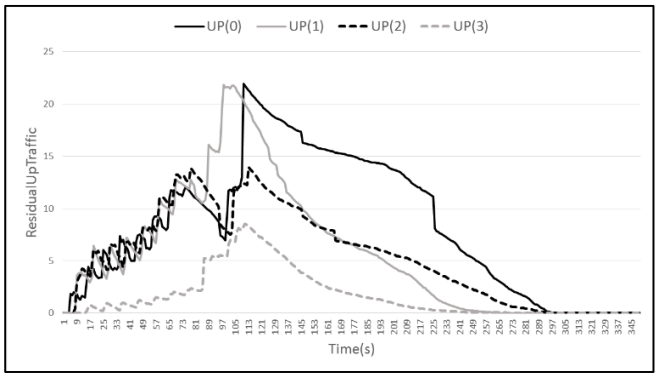


Figure 18. ResidualUpTraffic (Residual)

In summary, although it cannot cope with HDFS replication, the proposed Residual Scheduling can alleviate inter rack link load and shorten the duration of copy phase, thus achieving job performance improvement up to 23%.

C. Future Works

The proposed scheduler could not be fully compatible with replication procedure. This could be improved if we take into account the replication traffic in residual traffic. Also, in this paper we assumed sort jobs only, because it generates large amount of data transfer over the network. Mixture of other job types needs to be considered. Furthermore, implementation of proposed scheduling method needs to be studied.

VII. CONCLUSION

This paper discussed the impact of data transfer in Hadoop job performance. Using network simulation, it revealed the mechanism of job performance deterioration caused by the delay of copy phase due to the heavy load in the inter rack link of the cluster network. Based on this analysis, we proposed a new scheduling method -Residual Traffic Based Task Scheduling- that estimates the amount of inter rack data transfer in the copy phase and regulates task assignment accordingly. We evaluated this scheduling method and showed that the proposed method can improve Hadoop job performance significantly.

 ISBN: 978-1-61208-388-9

REFERENCES

[1] T. White. Hadoop: The Definitive Guide, 3rd Edition, O'Reilly Media / Yahoo Press,California, 2012.

[2] J. Dean, S. Ghemawat, "MapReduce: Simplified data processing on large clusters" Communications of the ACM 51.1 (2008): pp.107-113.

[3] The Apache Software Foundation. *Apache Hadoop.* [Online]. Available from: http://hadoop.apache.org 2015.1. 13.

[4] M. Zaharia, D. Borthakur, J. S. Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling". EuroSys conf., pp. 265-278, Paris, France, Apr. 2010.

[5] M. Zaharia, D. Borthakur, J. S. Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Job Scheduling for Multi-User MapReduce Clusters", Technical Report of EECS Department, University of California, Berkeley, 2009.

[6] A.Verma, B. Cho, N. Zea, I. Gupta, R. H. Campbell, "Breaking the MapReduce Stage Barrier", Cluster computing, 2013 – Springer

[7] J. Xie, S. Yin, X. Ruan, Z. Ding, Y.Tian, J.Majors, A. Manzanares, X. Qin, "Improving MapReduce performance through data placement in heterogeneous Hadoop clusters", IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), pp 1-9, 2010

[8] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, I Stoica, "Improving MapReduce Performance in Heterogeneous Environments", Technical Report of EECS Department, University of California, Berkeley, No. UCB/EECS-2009-183, Dec. 2009

[9] C. Qi, C. Liu, Z. Xiao, "Improving MapReduce Performance Using Smart Speculative Execution Strategy", IEEE Transactions on Computers, pp954-967, 2013

[10] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, I. Stoica, "Managing data transfers in computer clusters with orchestra" ACM SIGCOMM2011, pp98-109, 2011

[11] Amazon Web Services Inc. *Amazon Web Services, Cloud Computing: Compute, Storage, Datababase.* [Online]. Abailable from: http://aws.amazon.com 2015.1. 13.

[12] G. Wang, A. R. Butt, P. Pandey, K Gupta, "A simulation approach to evaluating design decisions in mapreduce setups." MASCOTS'09. Pp1-11, 2009.

[13] *ns-3*. [Online]. Available from: http://www.nsnam.org/ 2015.1. 13.

[14] H. Watanabe, M. Kawarasaki, "Impact of Data Transfer to Hadoop Job Performance - Architectural Analysis and Experiment -, ACMSE2014, Mar. 2014

[15] T. Chao, H. Zhou, Y. He, L. Zha, "A dynamic mapreduce scheduler for heterogeneous workloads." Grid and Cooperative Computing, 2009. GCC'09. Eighth International Conference on. IEEE, 2009.

[16] F. Ahmad, S. T. Chakradhar, A. Raghunathan, T. N. Vijaykumar, "Tarazu: optimizing mapreduce on heterogeneous clusters." ACM SIGARCH Computer Architecture News. Vol. 40. No. 1. ACM, 2012.

 ISBN: 978-1-61208-388-9

A Comparison Study of Information Security Risk Management Frameworks in Cloud Computing

Mohammed Alnuem
Information Systems Department
King Saud University
Riyadh, Saudi Arabia
Email: malnuem@ksu.edu.sa

Hala Alrumaih, Halah Al-Alshaikh
Information Systems Department
Al Imam Mohammad Ibn Saud Islamic University
Riyadh, Saudi Arabia
Emails: {hala.alrumaih, halah.al-alsheikh}
@ccis.imamu.edu.sa

***Abstract*—Nowadays, cloud computing is an emerging concept in the Information Technology (IT) industry. Cloud computing is not a new technology, but is a new way of using or delivering resources. It also enhances the efficiency of computation by providing centralized databases, memory processing and on demand network access. Consequently, cloud computing has become an important platform for companies to build their infrastructures upon. It allows organizations to use Internet-based services so that they can reduce start-up costs, lower capital expenditures, use services on a pay-as-you-use basis, access applications only as needed, and quickly reduce or increase capacities. If companies are thinking of taking advantage of cloud-based systems, they will have to seriously re-assess their information security risk management strategies. In recent years, numerous risk management frameworks based on information security have been proposed to manage the risk of cloud computing. This paper discusses how information security risk management is related to the cloud computing environment. It also presents seven different information security risk management frameworks that cover all of cloud service models and deployment models. These frameworks are classified according to coverage area of the framework. Moreover, the paper sheds light on some suggestions that may help cloud users. These suggestions are related to information security risk management in cloud computing and were obtained through a comparison made in this paper between the presented frameworks.**

Keywords-Cloud Computing; Risk Management Framework; Information Security.

I. INTRODUCTION

Recently, cloud computing has gained extensive attention, but the trust and security issues of cloud computing have prevented businesses from fully accepting cloud platforms. The security risks are associated with each cloud delivery model, cloud architecture and security controls involved in a particular cloud environment [1]. One security assessment tool that can reduce the threats and vulnerabilities and mitigates security risks is a risk management framework [2].

Conducting information security risk management is the core element of an Information Security Management System (ISMS). ISO 27001 is the international best practice standard for ISMS [3]. ISO/IEC 27000 provides policies, standards, guidelines and procedures for initiating, implementing, maintaining and improving information security management within an organization [4]. An information security policy consists of roles, responsibilities and defining the scope of information that must be protected across the cloud [5]. Standards ensure security consistency across the cloud and usually contain security controls relating to the implementation [5]. Guidelines consist of recommended, non-mandatory controls that help and support standards. Procedures consist of step-by-step instructions to assist workers in implementing the various policies, standards and guidelines [5]. Risk management frameworks can be based on one or more of the ISO 27001 information security framework requirements.

This paper focuses on presenting some information risk management frameworks for better understanding the critical areas in the cloud computing environment. The rest of the paper is structured as follows: Section 2 presents an overview of cloud computing. Section 3 introduces information security risk management features. Section 4 reviews seven different information security risk management frameworks for cloud. Section 5 shows a comparison between the risk management frameworks, while Section 6 concludes this study. while Section 6 concludes this study.

II. OVERVIEW OF THE CLOUD AND ITS FEATURES

Cloud computing moves the tasks and the resources from a local computer onto the larger computing center, which is shared among a large number of users and distributed on the Internet. The cloud system resource is transparent, as neither the application nor the user knows the location of the resource. Cloud resources are provided as a service on an as needed basis. Moreover, the user can decide to only pay for what they use [6]. Cloud computing include five key characteristics of on-demand self-service, ubiquitous network access, location independent resource pooling, rapid elasticity, and measured service [7]. According to Buyya et al. [8] the cloud computing has the following definition "Cloud is a parallel and distributed computing system consisting of a collection of inter-connected and virtualized computers that are dynamically

 ISBN: 978-1-61208-388-9

provisioned and presented as one or more unified computing resources based on Service-Level Agreements (SLA) established through negotiation between the service provider and consumers [8]." Vaquero et al. stated that "clouds are a large pool of easily usable and accessible virtualized resources (such as hardware, development platforms and/or services) [9]", where the resources can be dynamically reconfigured to adjust to a variable load (scale), allowing also for an optimum resource utilization. There are three cloud delivery models, as follows: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [6][7].

- *Infrastructure as a Service (IaaS):* Providing a working environment as per user wish and demand without manages or controls the infrastructure they simply control the storage and applications. One IaaS example is Amazon Elastic Compute Cloud (EC2).
- *Software as a Service (SaaS):* Cloud providers offer application software as on-demand services.. Examples of SaaS are Flickr, Google Docs, Siri, Amazon and Cloud Drive.
- *Platform as a Service (PaaS):* Provides infrastructure where you can create new applications. Consumer deploys their applications on the cloud computing system and controls their applications but they do not manage servers and storage. Examples of PaaS are Google App Engine, Amazon Web services.

Cloud computing is further divided into private cloud, public cloud and hybrid cloud, according to the different deployment models [6][7].

- *Private Cloud:* Private cloud is deployed for a particular organization and security can be created easily. Private clouds are virtualized cloud data centers inside a firewall. Private cloud refers to internal data centers of a business or other organization not made available to the general public.
- *Public Cloud:* Public cloud runs on the Internet and security is very complex. Public clouds are virtualized data centers outside of the firewall and resources are available to the consumer on demand over the public Internet.
- *Hybrid Cloud:* Hybrid cloud is a composition of two or more clouds and that are bounded by standard or proprietary technology. Hybrid clouds combine the character of both public and private clouds.

A new type of cloud deployment models offers Cloud Computing resources and services to mobile devices called Mobile Cloud Computing. Khan et al. defined mobile cloud computing as "an integration of cloud computing technology with mobile devices to make the mobile devices resource-full in terms of computational power, memory, storage, energy, and context awareness [10]".

However, in cloud computing the security situation is very different as the hardware, software and application data can be deployed and stored by the cloud providers. Therefore, to solve security problems that are occur in cloud computing there are some risk management frameworks can be used to secure the data transmitted, ensure the integrity of the applications and increase trust between users and service providers, etc. [11].

III. INFORMATION SECURITY RISK MANAGEMENT

Recently, cloud computing has gained considerable attention. Due to the involvement of many technologies including networks, databases, operating systems, resource scheduling, transaction management, concurrency control and memory management, various security issues have been highlighted as arising in cloud computing [12]. Indeed, security remains a major roadblock for organizations looking to reap the cost and efficiency benefits of the cloud. Table I summarizes cloud computing features and their corresponding security implications [13].

TABLE I. SECURITY IMPLICATIONS OF CLOUD FEATURES

Feature	Security Implication
Outsourcing	Users may lose control of their data. Cloud providers may use customers' data in a way that has not been agreed upon in the past.
Extensibility and Shared Responsibility	There is a tradeoff between extensibility and security responsibility for customers in different delivery models.
Virtualization	There needs to be mechanisms to ensure strong isolation, mediated sharing and communications between virtual machines.
Multi-tenancy	Issues like access policies, application deployment, and data access and protection should be taken into account to provide a secure multi-tenant environment.
Service Level Agreement	The main goal is to create a negotiation mechanism for the contract between providers and consumers of services.
Heterogeneity	Different cloud providers may have different approaches to providing security and privacy mechanisms. This will generate integration challenges.

As more organizations start to move their IT operations to the cloud, risk management remains a top concern. Risk management is the systematic application of management policies, procedures and practices to the tasks of establishing the context, identifying, analyzing, evaluating, treating, monitoring and communicating risk. It allows IT managers to balance the operational and economic costs of protective measures and achieve gains in mission capability

 ISBN: 978-1-61208-388-9

by protecting the IT systems and data that support their organizations' missions. Risk management encompasses three processes: risk assessment, risk mitigation and risk evaluation [14].

Risk assessment is the determination of a quantitative or qualitative output from risk analysis process [2]. It has four major processes. Likelihood determination process indicates the probability vulnerability that may be exercised within the construct of the associated environment. Impact analysis process determines the adverse impact resulting from a successful threat exercise of vulnerability. Risk determination process finds the risks and opportunities that impact on the associated environment using risk exposure formula and matrix. Control recommendations process provides the process and recommend controls that could mitigate or eliminate the identified risks. Risk mitigation refers to prioritizing, implementing and maintaining the appropriate risk-reducing measures recommended from the risk assessment process [14]. Cloud provider must develop Risk Treatment Plans (RTP) with multiple options (avoidance, transfer, retention, reduction and acceptance) [2]. The outcomes of RTP should be incorporated into service agreements because different models of cloud computing have various ways to mitigate vulnerabilities and threats. Risk evaluation is a continual process for implementing a successful risk management program [14]. It initiates specific follow-on actions as part of a comprehensive continuous monitoring program.

Well-planned risk management activities will be crucial in ensuring that information is simultaneously available and protected [15]. A well-structured risk management framework, when used effectively, can help management identify appropriate controls for providing the essential security capabilities to protect users' information, which is of high importance.

IV. INFORMATION SECURITY RISK MANAGEMENT FRAMEWORKS IN CLOUD

This section provides a review of seven different information security risk management frameworks for cloud computing which can be used to secure the data transmitted, ensure the integrity of the applications and increase trust between users and service providers. The information security risk management frameworks in this study have been classified according to the coverage area of the framework, and fall into the following: information security risk management frameworks for security evaluation in cloud environments, analyzing the risks in cloud environments and frameworks based on security policies.

A. Cloud Environments Security Evaluation Frameworks

Zhang et al. [2] have presented an information risk management framework for better understanding critical areas of focus in the cloud computing environment, to identify a threat and vulnerability by means of Deming Cycle. It is covers all of cloud service models and deployment models. The framework has seven processes, including: selecting relevant critical areas, and strategy and planning under the architecting and establishing the risk management program (PLAN) phase. Risk analysis, risk assessment and risk mitigation under the implement and operate (Do) phase, and then the assessing and monitoring program, and risk management review under the monitoring and review (Check, Act) phase. In selecting relevant critical area process, the authors enumerated twelve domains as areas of concern for cloud computing. At least one critical area that is relevant has to be selected before moving to the next process. The strategy and planning process is performed in collaboration with selecting relevant critical area to ensure plans effectively identify critical areas of focus and provide management with clear choices for resource allocation and optimization. Risk analysis process allows management to examine all currently identified threat and vulnerability concerns. Risk assessment is the following step that has four major processes–likelihood determinations, impact analysis, risk determination and control recommendations. Cloud providers must develop (RTP) with multiple options (avoidance, transfer, retention, reduction and acceptance) through risk mitigation process. The organization subsequently initiates specific follow-on actions as part of a comprehensive continuous monitoring program. Finally, a review is performed to develop a program effectiveness grid that is can be used first to establish a baseline, then set goals and objectives and evaluates progress.

However, the framework of Xie et al. in [11] pays more attention to how to increase the trust between users and service providers. This framework is composed of five basic processes: user requirement self-assessment, cloud service providers desktop assessment, risk assessment, third-party agencies' review and continuous monitoring [11]. At the user requirement self-assessment phase, the user should determine the required cloud computing model and security level. In the desktop assessment, the historical security status should be analyzed as should the potential risk of cloud service providers. The risk assessment of cloud provider includes seven stages: the preparation of risk assessment, asset identification, threat identification, vulnerability identification, existing security measures, risk analysis and risk assessment documentation. The third-party agencies review stage it is necessary to employ third-party agencies to review the procedure and to ensure security of cloud services. Through the previous stages, there is a need for a continuous monitoring process to monitor the ongoing risk assessment [11].

Popa et al. [16] explained the security issues related to private data and mobile cloud applications in detail. They proposed a mobile computing applications security framework called Secure Mobile-Cloud (SMC). It had to fulfill the following features: to make sure that the security of data is achieved when it is transmitted between the components of the same mobile application, also it had to verify the integrity of the applications either at the time of installation or updating on the mobile device. This solution takes into consideration the following constraints: mobile device energy, data sensitivity and users' options. The proposed framework best fits into SaaS layer of the cloud service delivery model by providing security services such

 ISBN: 978-1-61208-388-9

as confidentiality and integrity. SMC framework has several components running in the cloud and on the mobile. There are five kinds of managers: Mobile Manager, Mobile and Cloud Security Manager, Optimization Manager, Application Manager and Policy Manager, where each manager has a well-defined functionality. The framework also contains also the security components deployed on both cloud and mobile devices.

B. Cloud Environments Security Analysis Frameworks

Tanimoto et al. [17] identified various risk factors from a user's viewpoint by using the Risk Breakdown Structure (RBS) method. The risk analysis method is based on a risk matrix. Tanimoto et al. proposed a risk management framework that classified the risks into risk transference, risk mitigation, risk acceptance and risk avoidance, then categorized the problems in cloud computing according to these four classifications. Since risk transference problems tend to come from the cloud service provider, risks classified into risk mitigation tend to involve regulatory compliance of the cloud service provider, such as specification, authentication, etc. Risks in risk acceptance tend to be based on external factors, such as laws, while the risks in risk avoidance tend to be caused by different specifications of the cloud service provider and users [17].

Alhomidi and Reed [18] presented a resource independent framework for security risk analysis as a service (SRAaaS) suitable for IaaS model of cloud computing. The results of the analysis recommend a range of plans that can be used directly by cloud users to enforce the protection of their Virtual Machines (VMs), as the fundamental unit of IaaS, against possible attacks. The resource independence allows the SRAaaS to scan any kind of resources in the IaaS cloud so the framework can be extended to provide analysis for other types of IaaS resources, such as networks or storage. In addition, the framework reduces the need to consistently access the VMs or interrupt the processes and services running on each VM because time-consuming tasks in the risk analysis can be performed offline in the cloud. SRAaaS framework consists of seven steps [18] and only the first step has to be accomplished online while the further steps are performed offline. VM Vulnerability Scanning step aims to check all software and applications as well as the VM's connections and ports. Attack Graph Generation step visualizes how an attacker could exploit the VM by showing the attack paths generated using an attack graph generation tool. Risk Assessment Model step analyzes the produced attack graph by computing the most likely attacks, the highest-risk attacks and the highest loss attack. Security Control Selection step aims to help cloud users or providers with the selecting suitable and cheapest security controls that protect the VM from threats. Genetic Algorithm Optimization step simplifies the full attack graph to a smaller graph representing the most critical attack path. Analysis Results step describes potential vulnerability threats so the cloud user has an overall view of the existing security risks. Finally, Security Recommendations step reports a group of recommendations for the cloud user to make appropriate decisions regarding the security of the VM, including a list of required security controls and the total cost of the security controls as well as the cost of each control.

C. Frameworks Based on Security Policies

SecureCloud has been proposed by Takabi et al. [7]. It is a comprehensive security framework for cloud computing environments that preserves cloud security using identity management models and access control models. The framework consists of various modules to handle security and trust issues of cloud computing environments. The modules deal with issues such as access control to provide the security and privacy specification and enforcement functionality, policy integration among multiple clouds to integrate access policies of different policy domains and define global access policies, secure service that is responsible for secure service discovery, composition and provisioning. In addition, these modules cover issues like trust management, which is responsible for negotiation, establishment, and evolution of bidirectional trust between different clouds and between a cloud and its users, semantic heterogeneity to check the correctness of the integrated policies among policies from different clouds and identity management which is responsible for authenticating users and services based on credentials and characteristics.

Zhao [4] introduced a risk management framework based on aligning policies relating to organization's IT policies and standards and security management to fit with the cloud computing model. The security solutions provider may need some standards and guidelines to evaluate the cloud service provider against regulatory requirements. Some of the standards, frameworks and guidelines such as ITIL, Statement on Auditing Standards (SAS), ISO/IEC 27000 standard series, Control Objectives for Information and Technology (COBIT) framework, Data Security Standard (PCI DSS), Cloud Security Alliance Cloud Controls Matrix (CCM) and others. This framework found that to build a model of security management in any cloud service provider, it is necessary to start by identifying the asset for the cloud deployment then evaluating the risk for the asset [4].

V. DISCUSSION

In this section, the previous information security risk management frameworks are compared according to the ISO 27001 framework, and the classification that was presented previously in this study.

The comparison based on different indications such as cloud service models, deployment, framework concentration, and determine which ISO 27001 requirements the framework based on.

A. Cloud Environments Security Evaluation Frameworks

By examining the prior cloud environments security evaluation frameworks, it is clear that the frameworks of Zhang et al. [2] and Xie et al. [11] focus on user perspectives to evaluate cloud environment security. The two frameworks can fit on all of cloud service models and deployment models. Both frameworks concentrate on

 ISBN: 978-1-61208-388-9

denoting where the user should specify the process information of the cloud computing service, and determining the necessary cloud computing model as well as security level, at the first stage in the their frameworks. In Xie et al.'s framework, before assessing the risk of service provider, users have to evaluate cloud service providers' plans, analyze the historical security status and, furthermore, acquire the potential risk of the cloud service providers. The risk assessment in the framework of Zhang et al. has two processes, risk analysis and risk assessment to identify the threats and vulnerabilities then determine the likelihood that a potential vulnerability could be applied and its impact. In addition both frameworks have monitoring process. As part of assessing cloud provider, specific follow-on actions are initiated for a continuous monitoring program of effectiveness after the RTPs are implemented. According to the framework requirements in ISO 27001, both frameworks implement information security management based on procedures. Moreover, the Zhang et al. framework was developed in a standard quality management (PDCA) cycle of continuous improvement, based on evolving standards of ISO 27001. The other framework that evaluates cloud environment security works on a mobile cloud application called Secure Mobile-Cloud (SMC) [16]. This framework fits in the SaaS layer only. SMC framework aims to secure data communication between the same application components. In SMC framework, there are five different kinds of managers and security components on the mobile side and cloud side. All of these managers have their own functions to analyze and evaluate risks. This framework is also based on procedures according to the framework requirements in ISO 27001.

B. Cloud Environments Security Analysis Frameworks

Regarding the presented cloud environment security analysis frameworks, both Tanimoto et al. [17] and Alhomidi and Reed [18] found that cloud computing security has not been sufficiently investigated, although cloud computing services have. As a result, they developed a number of frameworks capable of analyzing the security risks and extracting of risk factor in the cloud computing environment. In [17], the authors analyzed and extracted risks of utilizing cloud computing by using the Risk Breakdown Structure (RBS) method. RBS is a typical risk Aanalysis method of the project management method. Meanwhile, SRAaaS framework [18] provided a mechanism for risk analysis using attack graph, which is an important tools used to present the relationships between vulnerabilities. Furthermore, Tanimoto et al. started the analysis process by identifying various risk factors from a user's viewpoint, then classified risks using risk matrix method that classified risks into four kinds risk avoidance, risk mitigation, risk acceptance, and risk transference. Finally, it developed countermeasures individually to satisfy extracted risks and then detailed the risk management proposals for each classification. On the contrary, SRAaaS framework, which is suitable for IaaS model of cloud computing, consists of seven steps. It starts with vulnerability scanning step to check all software and applications, as well as connections and ports. Then it ends with the security recommendations step that reports a group of recommendations for the cloud user in order to make appropriate decisions regarding the security; this includes a list of required security controls and the total cost of the security controls as well as the cost of each control. The framework in [17] is based on procedures according to the framework requirements in ISO 27001. Alternatively, SRAaaS framework recommends a range of guidelines that can be used by cloud users to enforce the protection of their services and systems against possible attacks, according to the framework requirements in ISO 27001.

C. Frameworks Based on Security Policies

Looking into the introduced frameworks that are based on security policies, Takabi et al. [7] and Zhao [4] evolved frameworks based on aligning various policies and standards relating to information security risk management frameworks, in order to fit with the cloud computing model. SecureCloud framework [7] was built based on the existing research on multi-domain policy integration and the secure service composition. The framework consists of various modules to handle the security and trust issues of cloud computing environments. The important module in SecureCloud framework is policy integration in the cloud that should be able to address challenges such as semantic heterogeneity, secure interoperability and policy evolution management. On the other hand, Zhao's framework supports standards aligning of one cloud service provider. It found that the building of a model of security management in any cloud service provider starts with identifying the asset for the cloud deployment then evaluating risk for the asset. Accordingly, both frameworks are based on policies according to the framework requirements in ISO 27001.

The following Table II represents a summary of the comparison of the previous information security risk management frameworks:

 ISBN: 978-1-61208-388-9

TABLE II. SUMMARY OF THE COMPARISON BETWEEN SOME INFORMATION SECURITY RISK MANAGEMENT FRAMEWORKS

Criteria / Framework Name	Coverage area	Cloud service and deployment models	Concentration	Framework Requirements in ISO 27001
Zhang et al. [2]	Cloud Environments Security Evaluation	Fit in all of cloud service and deployment models	Identify the threats and vulnerabilities and its impact of using the cloud.	Based on procedures and standard
Xie et al. [11]	Cloud Environments Security Evaluation	Fit in all of cloud service and deployment models	Users have to evaluate cloud service providers' plans, analyze the historical security status and, acquire the potential risk of the cloud service providers.	Based on procedures
SMC framework[16]	Cloud Environments Security Evaluation	Fits in the saas layer	Have five different kinds of managers and security components on the mobile side and cloud side.	Based on procedures
Tanimoto et al. [17]	cloud environment security analysis	Fit in all of cloud service, deployment models	Analyzed and extracted risks of utilizing cloud computing by using the Risk Breakdown Structure (RBS) method. And using risk matrix method that classified risks into four kinds of risks.	Based on procedures
Alhomidi and Reed [18]	cloud environment security analysis	Fits in iaas model.	Provided a mechanism for risk analysis using attack graph to present the relationships between vulnerabilities.	Based on guidelines.
Takabi et al. [7]	Based on Security Policies	Fit in all of cloud service, deployment models	Handle the security and trust issues of cloud computing environments by using various modules.	Based on policies.
Zhao [4]	Based on Security Policies	Fit in all of cloud service, deployment models	Identifying the asset for the cloud deployment then evaluating risk for the asset.	Based on policies.

VI. RESULT

After reviewing seven information security risk management frameworks, this study makes some suggestions related to information security risk management in cloud computing. The organizations that have decided to move to cloud computing have to define the benefits and risks of cloud computing and implement processes to manage security risk. The information security risk management policies should comply with an organization's IT policies and standards to protect the confidentiality, integrity and availability of information security. The study observes some of the main processes that are needed to manage security risks. The first step in risk management is

 ISBN: 978-1-61208-388-9

assessment of the cloud service provider to evaluate the cloud service plans and analyze the historical security status. The second step entails a risk analysis of cloud provider by identifying the assets, threats and vulnerabilities. Risk assessment is the third process in the implementation of risk management. Risk assessment means assessing security incidents from two dimensions, i.e., the likelihood and the adverse impact of an incident. After the risk management plans are implemented, there is a need for follow-on actions as part of a comprehensive assessment and continuous monitoring program for effectiveness. Moreover, the study acquires different recommendations that may help cloud providers and customers with choosing between the existing information security risk frameworks. It is important to specifying the cloud service models and deployment models, the purpose of the framework, and whether there is a need for one or more cloud environments/providers. Moreover, there is a need to specify the framework requirements based on the basic framework requirements in ISO 27001: policies, standards, guidelines and procedures.

VII. CONCLUSION

Cloud computing has the advantages of high efficiency and low costs, as well as scalability. As the march of cloud computing continues, security issues have appeared. Indeed, the investigations in the cloud computing environment have mainly focused on the service side, while the security side has not been sufficiently looked at. This study presented and compared seven different information security risk management frameworks that covered all of cloud service models and deployment models. The comparison was undertaken according to the framework requirements in ISO 27001 and the coverage area of the frameworks that were presented in this study. Moreover, the paper sheds light on some suggestions related to the information security risk management in cloud computing.

REFERENCES

[1] CPNI Centre for the Protection of National Infrastructure, "Information Security Briefing 01/2010 Cloud Computing," pp. 36, March 2010.

[2] X. Zhang, N. Wuwong, H. Li, and X. Zhang, "Information Security Risk Management Framework for the Cloud Computing Environments," 2010, pp. 1328–1334.

[3] ISO/IEC 27001 - Information security management, retrieved from: http://www.iso.org/ accessed in 4/5/2014

[4] G. Zhao, "Holistic framework of security management for cloud service providers," in *Industrial Informatics (INDIN), 2012 10th IEEE International Conference on*, 2012, pp. 852–856.

[5] P. Johnson, "What are Policies, Standards, Guidelines and Procedures? | MindfulSecurity.com – The Information Security Awareness Resource." accessed in 20/4/2014

[6] W. Liu, "Research on cloud computing security problem and strategy," in *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, 2012, pp. 1216–1219.

[7] H. Takabi, J. B. D. Joshi, and G.-J. Ahn, "SecureCloud: Towards a Comprehensive Security Framework for Cloud Computing Environments," Computer Software and Applications Conference Workshops (COMPSACW), 2010 IEEE 34th Annual , pp. 393–398.

[8] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, Jun. 2009, pp. 599–616.

[9] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, 2009, pp. 50–55.

[10] A. U. R. Khan, M. Othman, S. A. Madani, and S. U. Khan, "A Survey of Mobile Cloud Computing Application Models," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, 2014, pp. 393–413.

[11] F. Xie, Y. Peng, W. Zhao, D. Chen, X. Wang, and X. Huo, "A risk management framework for cloud computing," in *Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on*, 2012, vol. 1, pp. 476–480.

[12] K. Hamlen, M. Kantarcioglu, L. Khan, and B. Thuraisingham, "Security Issues for Cloud Computing:," *International Journal of Information Security and Privacy*, vol. 4, no. 2, 2010, pp. 36–48.

[13] H. Takabi, J. B. D. Joshi, and G.-J. Ahn, "Security and Privacy Challenges in Cloud Computing Environments," Security & Privacy, IEEE Nov.-Dec. 2010, vol.8, no.6, pp.24,31.

[14] S. R. Vallabhaneni, *Corporate Management, Governance, and Ethics Best Practices*. John Wiley & Sons, 2008.

[15] ISACA, "Cloud Computing: Business Benefits with Security, Governance and Assurance Perspectives," An ISACA Emerging Technology White Paper, pp 7.

[16] D. Popa, M. Cremene, M. Borda, and K. Boudaoud, "A security framework for mobile cloud applications," in *Roedunet International Conference (RoEduNet), 2013 11th*, 2013, pp. 1–4.

[17] S. Tanimoto, M. Hiramoto, M. Iwashita, H. Sato, and A. Kanai, "Risk Management on the Security Problem in Cloud Computing," in Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference, pp.147,152.

[18] M. A. Alhomidi and M. J. Reed, "Security risk analysis as a service," in *Internet Technology and Secured Transactions (ICITST), 2013 8th International Conference for*, 2013, pp. 156–161.

 ISBN: 978-1-61208-388-9

Associating Performance Measures with Perceived End User Performance:

ISO 25023 compliant Low Level Derived Measures

Anderson Ravanello, Luis Villalpando, Jean-Marc Desharnais, Alain April, Abdelouahed Gherbi
Department of Software Engineering and Information Technologies
École de Technologie Supérieure, University of Quebec,
Montreal, QC, Canada
Email: ravanello@gmail.com, luis.bautistav@gmail.com, jean-marc.desharnais@etsmtl.net,
alain.april@etsmtl.ca, Abdelouahed.Gherbi@etsmtl.ca

***Abstract*— This paper applies a measurement procedure to predict the degraded state of a private cloud application using only available data center log low level derived measures (LLDM). Our intent is to improve the discussion of service level agreements of a widely used private cloud computing application (i.e. 80,000 users on 600 servers world-wide). In organizations, cloud application performance measuring is often based on subjective and qualitative measures with very few researches to address the large-scale private cloud perspective. Furthermore, measurement recommendations from ISO proposals (i.e. ISO 250xx series, ISO/IEC 15939 and more recently the ISO/SC38-SLA framework) are poorly adopted by the industry, mainly due to the absence of proof of concept and the high degree of complexity associated with implementing the measurement concepts described in these international standards. To try to demonstrate these concepts, the ISO 25010 performance efficiency characteristics are used with a number of LLDMs to model the state of a large private cloud computing application using indicators such as: normal, abnormal, adequate or degraded. This application still cannot be generalized due to its nature as research in progress.**

Keywords- cloud computing; cloud application; SLA; ISO 25010; end user performance measurement

I. INTRODUCTION

Measuring end user performance has been a concern of software engineering researchers since the early 60's [1]. Many experiments have been created, tested and validated [2][3][4] based on a survey with the involved users. Surveys have concerning limitations, such as not being good for following trends in real time, not providing good source for cause and effect, having poor timing response, demonstrating low response and being vulnerable to responder bias. To avoid these frailties, some form of automated, user-independent approach would be helpful.

Software systems performance measurement is currently conducted in many ways. One popular approach is to use the data center logs readily available in different operational systems, applications, computers and IT (Information Technology) infrastructure components. Logs are binary files that collect data from different components in a system and store this data in a file or database for posterior analysis. Many commercial, open source, and easily accessible tools, are available for collecting, analyzing and generating performance dashboards that present technical measures (Low Level Derived Measure - LLDM) of different system components that are used by a software[5],[6],[7].

Measuring performance using measures issued from logs can only measure the internal, and very technical, perspectives of an IT system. This is why the end user - typically the actor who uses the systems for daily activities - performance perspective is often inferred, estimated, approximated and even sometimes guessed, based on experience and using data center log data. The resulting measures may affect or not the actual user's perceived performance according to the observer's perspective and experience [8], [9], [10].

Cloud computing operates in complex environments which are dependent on a number of IT infrastructures, including components that are often widely geographically dispersed, with shared elements and running diverse applications[11]. This technology uses hardware and software to deliver ubiquitous, resilient, scalable, billed-by-use, agnostic application systems [12]. There are many advantages [13], [14] and disadvantages [15] to using such a technology, and one of its major disadvantages contemplated in this research is: the unreliable system performance due to the complexity of the infrastructure used by cloud applications.

Considering these challenges, the authors approach this case study with one particular hypothesis: Is it possible to employ existing data center logs to model degraded cloud computing application performance via the monitoring of two sources of data: 1) low level derived measures and 2) the end user's reports to help desk of such degradations?

In order to conduct this case study, the authors follow three methodological steps:

1. Associate user reports of degraded performance with the LLDM;
2. Map the base and low level measures to the quality characteristics proposed by Bautista[16];
3. Perform a lab experiment to model the performance of a real cloud application.

This paper follows the following structure: On section II, III and IV, a bibliographical review is performed on the subjects of End User Performance Measurement, the ISO 25000 standard and cloud computing, respectively. Sections V, VI, VII present the case study, the challenges and

 ISBN: 978-1-61208-388-9

conclusion, as well as the next steps of this research, also respectively.

II. END USER PERFORMANCE MEASUREMENT

The problem of measuring information system's performance is not new and has been explored by numerous authors. A number of these use available data center tools for measuring the values of low level and derived measures. Different approaches are implemented in available tools: some install agents on the involved nodes that report the measures back to the performance management database [17]; others monitor the measures via SNMP (Simple Network Management Protocol) [18], collecting the measurements directly and other tools store the measurements locally on performance logs [5]. These measures are usually processed locally for monitoring purposes or stored and processed for later analysis.

III. ISO 25000 SOFTWARE QUALITY ASSESSMENT STANDARD

A. The ISO 25000 Family of standards (SQuaRE)

The Software product Quality Requirements and Evaluation, (SQuaRE) series of standards is composed of many documents destined for many different audiences. It is made up of 5 groupings and 14 documents, the most important of which are shown in Figure 1: Quality Management (ISO/IEC 2500n), Quality Model (ISO/IEC 2501n), Quality Measurement (ISO/IEC 2502n), Quality Requirements (ISO/IEC 2503n), Quality Evaluation (ISO/IEC 2504n) and its Extensions (ISO/IEC 25050 - 25099).

B. ISO 25010 characteristics and sub characteristics

ISO 25010 describes three different quality models for software products: 1) quality-in-use model; 2) product quality model; and 3) data quality model. Each of these models proposes different quality characteristics to represent the quality concepts required to assess software performance from the various perspectives. The first of these, the quality-in-use model, which is designed to measure the quality of software from a user's perspective, proposes five characteristics: effectiveness, efficiency, satisfaction, freedom from risk, and context coverage. The second, the software product quality model, proposes eight characteristics: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. (We do not consider the data quality model in this paper.)

The main challenge in assessing quality in use and the quality of a software product is to answer the following questions:

-What are the best characteristics and sub characteristics for evaluating the quality of the system to be measured?

-Which derived measures will help in evaluating the quality of the system to be measured based on the characteristics and sub characteristics selected?

-Which measures can be used to form the basis of the derived measures?

In the next section, we explain the concepts of the base measure and the derived measure, as defined by the ISO.

C. Base and derived measure concepts (ISO 15939 and ISO 25021)

A base measure is "a measure defined in terms of an attribute and the method for quantifying it" and a derived measure is a measure that describes a function of two or more values of base measures [19], and are derived respectively from a **measurement method and a measurement function**. Identical definition is proposed in the International Vocabulary of Basic and General Terms in Metrology [20]. The quality measure elements (QME) are either a base or a derived measure, which means that a LLDM could be a QME [21, 22]. This definition is an adaptation of the one in the International Vocabulary of Basic and General Terms in Metrology [23]. The International Vocabulary of Basic and General Terms in Metrology (VIM) is the standard used to define unit measures in science (e.g. meter, degree Celsius, etc.). A measurement method is defined as a logical sequence of operations, described generically, which is used in quantifying an attribute with respect to a specified scale [23]. It is also based on the definition in [20]. A measurement function is defined as an algorithm or calculation that combines measures [25].

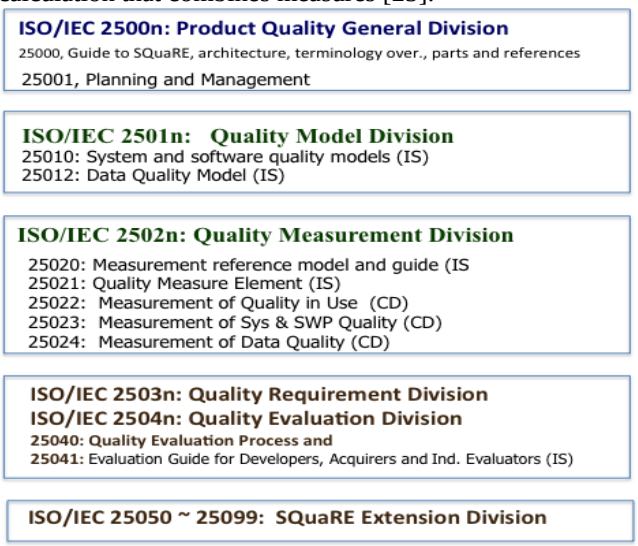


Figure 1: Five document groupings

IV. CLOUD COMPUTING

As we have presented in the introduction, cloud computing is a complex technology that depends on different infrastructures that include components that are often dispersed geographically, with shared elements and running diverse applications [26]. This technology employs hardware and software to deliver ubiquitous, resilient, scalable, billed-by-use, application agnostic systems [12]. In the scope of this research, the cloud-computing infrastructure analyzed fits the classification of a Private cloud.

One of the frequently cited sources for the definition of cloud computing is the US National Institute of Standards

and Technology (NIST), that proposes that "*Cloud computing is a model for enabling convenient, on demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [27].

Cloud computing is offered or assembled in different formats to the consumers. Three formats are the most prominent: Infrastructure as a Service (Iaas), Platform as a Service (PaaS) and Software as a Service (SaaS).These services can be deployed in different formats, mostly constraining cost, administrative effort, customization and privacy requirements, being Public, Private and Hybrid.

Infrastructure as a Service (IaaS) is an offer where a provider offers virtual or physical computing resources (CPUs, memory, disk space) over which a customer is free to deploy and manage its own environment. This allows a greater degree of customization, but causes a larger overhead in management processes to the client. Amazon Elastic Compute Cloud is one example of such a service [28].

Platform as a Service (Paas) is a different offer whereas a set of computing resources, operational systems and development tools are hosted by the provider and the customer is capable of creating services and applications that are compliant to the offer's characteristics, with a limited degree of customizability. This offers greater stability and control of computational resources, as the customer can focus on developing or hosting the products and services owned without having to spend resources in managing, updating and maintaining the infrastructure. One such offering is the Windows Azure Platform [29].

Software as a Service (SaaS) is a form of offer where the consumer accesses applications, services and information from a standard interface, having low customizability but no administrative effort. These applications are hosted and managed completely by the provider. One such application is the widely used Gmail application by Google [30].

Public Clouds are owned, managed, configured and controlled by the service providers who can then offer the cloud third party clients. **Private clouds are built for specific organizations**, with the possibility of outsourcing its management to third parties. **Hybrid clouds** contain one or more components that are owned by private and public parties.

SaaS Description of this case study: In this case, the evaluated SaaS is responsible for servicing e-mail clients, encompassing the desktop application, active directory authentication, network transport, message storage and indexing. The minimum system requirements are described in [31].

V. CASE STUDY

In order to address the research problem of the possibility of employing data center logs to model degraded cloud computing application performance via the monitoring of end user's reports of such degradations, the authors perform an exploratory case study where users complaints, in the form of incidents or trouble tickets, reported to a help desk, are studied during an specific work period. Whenever these trouble tickets relate to the studied SaaS application, the performance logs from the all the nodes represented on Figure 2 are collected in a performance management database. These performance measures undergo the 3 steps presented, at the end of Section I.

The following methodological protocol is applied during the case study:

Data Collection: Data is collected from two different sources: a) the Information technology Service Management system (ITSM) that is accessed and maintained by the help desk for record keeping and b) the data center logs collected. During the case study we received 30 complaints at the help desk and collected approximately 4 GB of datacenter logs for this application.

Data organization: For the help desk tickets, data is concentrated on the smallest time segment possible in order to represent the most amount of complains with as minimal environmental variation as possible. For the performance logs, three different work windows are open: 1) the moment of the degradation report at the help desk, 2) the previous three hours, and 3) the anterior week. After the data collection phase, the LLDM are associated to the ISO quality characteristics. Then, we conduct data analysis. Two distinct processes are used for analysing the data. First, the

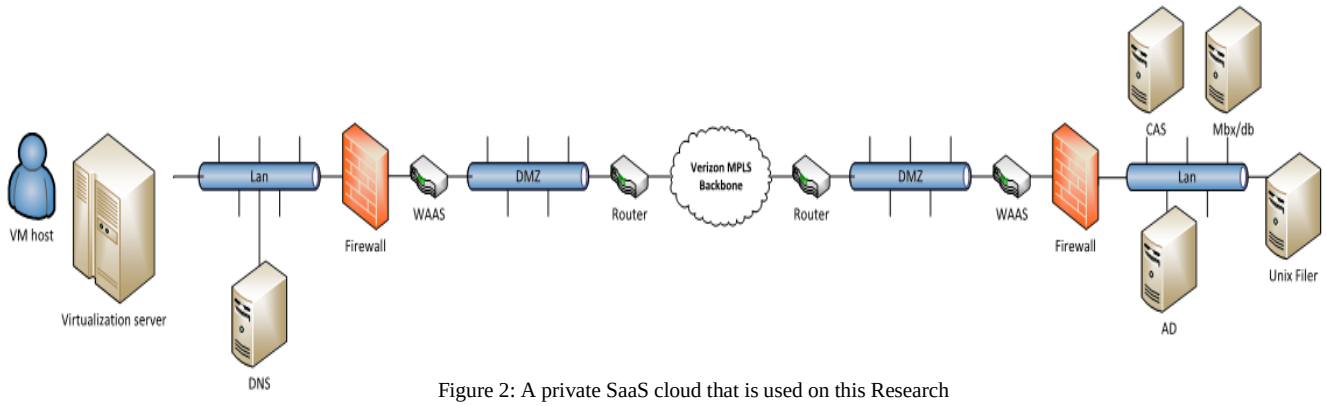


Figure 2: A private SaaS cloud that is used on this Research

ticket information is manually read to identify clearly the performance issues. Second, statistic data of the logs is compared for the three previous work windows as well as in between reports in order to identify similarities between the degradation reports. On this research, the logs employed follow a "RAW" format, whereas the performance data is represented by "Timestamp: LLDM name; LLDM Value". There is no metadata available on these logs that can help on accelerating indexing or searching the information.

Data interpretation is conducted in order to identify the possibility to map the user complaints to the LLDM's and then the LLDM's to the ISO quality characteristics, which would offer a method for monitoring LLDM's to 1) understand the user perspective; and, 2) generate quality indicators for the application under study.

Once the main measurement steps have been done, the following data processing is done.

Collection of user degradation reports: This involves monitoring an Information Technology Service Management (ITSM) system for new tickets, searching for keywords such as "e-mail", "slow", "slowness", and "hanging". For each complaint that matches the keywords, the machine name and time stamp are recorded. During this case study, 30 such cases were observed, covering 45 minutes.

Logs collection: For the 30 above cases, the relative data available has been collected for further analysis. This totals to 4GB of data organized initially in different files, then transformed in a NOSQL (Not Only Structured Query Language) file for statistical analysis. The NOSQL files are organized as the lines representing the time series events and each column being a different LLDM.

Association of the LLDM: This is an adaptation of quality characteristics of ISO 25023 standard.

Statistic exploration of the low level derived measures. The LLDM identified during step 2 are then compared, using covariance and correlation techniques, aiming to reduce the total amount of observed data. Then, skewness and kurtosis of the 3 work windows are calculated, in order to establish a baseline (week), and escalading scenario (for the three previous hours) and a reported event (the hour). This allows us to see if there is any difference between the baseline and the actual degradation report. Furthermore, principal component analysis (PCA) is calculated to determinate measures with the most impact. From the PCA, frequency and trend lines are determined for the values, in order to link the values back to the user reports of degradation (i.e. the Help desk tickets). This helps in identifying a) which measures have more significance and b) to which extent they affect the user experience. This step is ongoing as of the writing of this report.

VI. CURRENT RESULTS:

In this section, we present the results of each of the methodological steps and sub steps presented on section V [1-4], beginning by the presentation of the step-by-step approach of the execution of this case study:

1 – Identify degradation report: The tickets logged at the help desk are analyzed for the keywords (i.e. "slow", "hanging", and "slowness", amongst others). Tickets, which contain these keywords, are flagged as potential performance degradation issues.

2 – Data extraction and organization process: Extract the raw performance data associated with the performance degradation report, for 1 week of time. 63589 data points were collected, with 38 LLDM. We observe 33 high degree correlations (I.e: >+0.74), while 12 presented a strong negative correlation (i.e. <-0.60), from which we reduced the 38 initial measures to 15. These have been selected based on the described statistics approaches and also based on logical response of being regarded as being available when the value is lower; for example, Memory_Commited_Bytes is selected instead of Memory_Available_Bytes, mainly because both are strongly uncorrelated (i.e. -0.98) and because the smaller amount of committed bytes, more will be available. For this case study, the selected list of LLDM, named as per the according logs, is: %_Processor Time; Page_File_%_Used; Commited_Bytes; AVG_Disk_Read_Queue; I/O Read; Private_Bytes; Thread_Count; Handle_Count; AVG_Disk_Write_Queue; Connection_Failures; Pages/Sec; Connections_Active; Connections_Reset; Disk_Free_MB;

These LLDM are then associated to the Performance Measurement Framework Cloud Computing Concepts proposed by Bautista, resulting in the classification shown on Table I. For the indicators demonstrated in this paper, only Performance measures have been utilized as further described in Section VI.

Using this data organization, it is possible to plot the graphical representation of the ISO 25023 quality concepts as displayed on Figures 3 and 4.

Figure 3 represents one single data point in time, whereas Figure 4 represents the collection of all data points for one specific concept – Performance Efficiency: Resource Utilization.

Figure 4 is purposely left on this format to represent the challenge of interpreting all performance data for the cloud simultaneously. This is then comparable to the diagram presented on Figure 5.

Figure 5 represents the resource utilization indicator, demonstrating visible peaks on the observations 32, 5226, 11132, 22651, and 37022.

These observations correspond to the following time stamps: Monday 07:56 AM, Tuesday, 08:05 AM, Wednesday 10:28 AM, Thursday 07:53 AM and Friday 10:38 am. These peaks in resource utilization could point towards a pattern on resource utilization on those specific times of the day, possibly indicating a degraded performance from the perspective of this quality characteristic.

 ISBN: 978-1-61208-388-9

Upon reviewing 30 of such reports along with the steps that were taken for recovery, in 24 cases the technician giving the support observed either or a combination of 4 LLDM - processor, memory, disk space and network utilization – to determine the possible root cause for each case. In at least 6 of the cases, the user who first presented the complaint reported a recurrence of the issue.

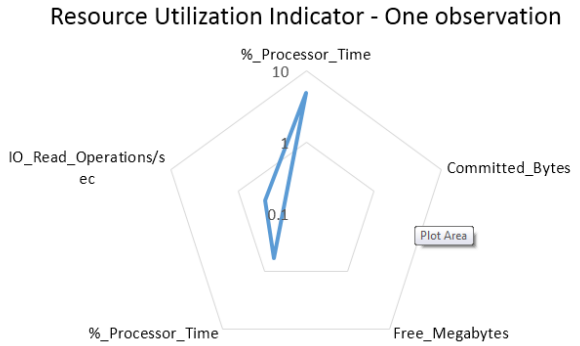


Figure 3 – Single observation of the resource utilization values

Eighty-four different LLDM exist on the logs relative to these 30 cases. Twenty-two are strongly correlated (+0.74) and 19 are strongly un-related. If the empirical model of 4 measures seems simplistic, the exploratory model of 84 measures can be optimized via correlation and covariance, for reduced measures. Utilizing population variance, 14 LLMD demonstrate higher comparative significance.

TABLE I. ASSOCIATION OF LLDM AND ISO 25023 CONCEPTS

Name of the Selected LLDM as extracted from the logs	**Concept to which the LLDM can be associated according to ISO 25023**
%_Processor Time Commited_Bytes Disk_Free_MB Process_%Processor_Utili I/O_Read Private_Bytes	Performance Efficiency – Resource Utilization
Page_File_%_Used Avg_Disk_Read_Queue Avg_Disk_Write_Queue Connections_Active Pages/Sec Handle_Count Thread_Count	Performance Efficiency – Capacity
Connections_Failures	Reliability – Maturity
Connections_Reset	Reliability – Fault Tolerance

For these LLMDs, the skeweness and kurtosis is calculated, generating a possible classification into generalizable (low kurtosis) and non-generalizable (high kurtosis).

By observing the values of the LLDM, it is possible to link positively the higher values with the complaints of the users, indicating that the empirical knowledge disclaiming that the lowest levels of utilization will provide better user experience. With the association of the measures in quality characteristics, it is possible to support the creation of quality indicators derived from LLDM that can represent the user perception of such derivations, possibly leading to an indicator of the service level of the cloud computing system

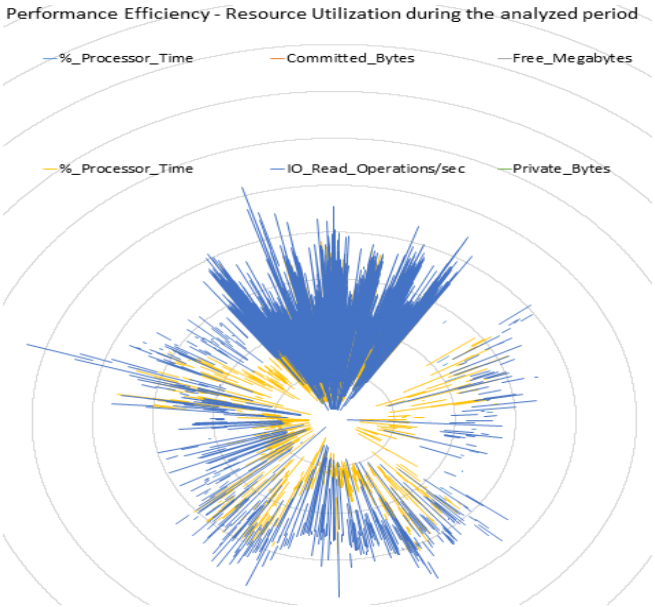


Figure 4 – Multiple observations of the resource utilization values

Figure 3 represents the collection of a single data point of the resource utilization for this application. Figure 4 illustrates the complexity of demonstrating all the data points in a single graph, which prompts the creation of the resource utilization indicator shown on Figure 5.

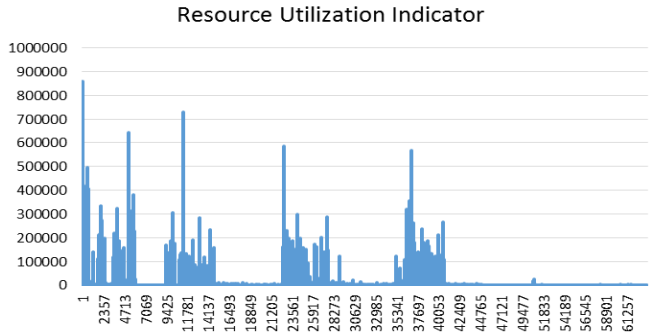


Figure 5 – Resource Utilization Indicator

VII. CHALLENGES AND CONCLUSION

During the planning of this case study, there was some expectation that the experimentation would be laborious because of the many manual processes involved with implementing the ISO quality standards concepts as well with the many statistics analysis involved:

-Data collection challenges: Reading through help desk tickets might not be the best method for extracting user reports of performance degradation. Issues about misspellings, synonyms, different technician interpretation all combined created, initially, some subjectivity. Additionally, extracting and transforming the data from the existing log tools format, which is stored in a relational database, to convert to a NO-SQL database format (i.e. in order to be able to perform the statistical analysis) was also a difficult process initially; whereas the data is exported

 ISBN: 978-1-61208-388-9

form one system in text format, then a NO-SQL table has to be designed and the data imported.

-Data analysis challenges: the recursive calculations necessary for the PCA and outlier detection require computing power. Performing these calculations, in real time, have proven to be a challenge.

-Data interpretation challenges: even though the statistical techniques help for reducing the amount of data giving quantitative data for decision making, there is still some subjectivity degrees involved in analyzing the data; the analyst's ability to interpret the data, especially in comparison to other data sets, may influence the results of the data analyses.

Despite these challenges, it was possible to a) manually associate the LLDM as per Bautista's framework, expanding the original research; b) statistically analyze the data in order to produce tentative indicators, promoting a better understanding of the end user performance. The method, the framework and the processes are still far from conclusive, as expected for a research in progress.

VIII. NEXT STEPS

As described, this short paper presents the results of a research still in progress. The following activities are underway: Automated data extraction and consolidation; Automation of the baselines, correlation and co-variation calculations; Automation of the data reduction; Frequency analysis for outlier detection of the performance values, which would strongly link the values with the end user perception of performance; Implementation of quality indicators

REFERENCES

[1] J. C. Emery, "The Impact of Information Technology on Organizations," 24th Annual Meeting, Academy of Management, pp. 1, 1964.

[2] R. Buyya, C. Yeo, S. Venugopal, J. Brober and I. Brandic, Vision, hype, and reality for delivering computing as the 5th utility. FGCS, Volume 25, I 6, 2009, pp. 599-616. 2009.

[3] S. &. W. S. Davis, " The mediating effects of intrinsic motivation, ease of use and usefulness perceptions on performance in first-time and subsequent computer users," Interacting with Computers, 13(5), pp. 549-580, 2001.

[4] J. F. A. Etezadi-Amoli, A structural model of end user computing satisfaction and user performance, ELSEVIER, Information & Management pp.30 - 65-73, 1996.

[5] Microsoft, "Microsoft Perfmon," 2013.[Online]. Available: http://technet.microsoft.com/en-us/library/cc749249.aspx. [Accessed 05/ 2014].

[6] Nagios, "Nagios IT Infrastructure Monitoring," 2013.[Online]. Available: www.nagios.org.

[7] J. Weisberg, "Argus TCP Monitor," 2013.[Online]. Available: http://argus.tcp4me.com. [Accessed 05/ 2014].

[8] Microsoft, "Performance Analysis of Logs Tool," 2012.[Online]. Available: http://pal.codeplex.com/. [Accessed 05/ 2014].

[9] A. Friedl and S. Ubik, Perfmon and Servmon: Monitoring Operational Status and Resources of Distributed Computing Systems, CESNET Technical report 1/2008, 2008.

[10] R. Kufrin, Measuring and improving application performance with Perfsuite, http://perfsuite.ncsa.illinois.edu/publications/LJ135/, 2005. [Accessed 05/ 2014].

[11] S. Sivathanu, Y. Mei, L. Liu and X. Pu. Performance Measurements and Analysis of Network I/O Applications in Virtualized Cloud, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.165.6742&rep=rep1&type=pdf, 2010. [Accessed 05/ 2014].

[12] B. R. Prasad, E. Choi and I. Lumb, A taxonomy, survey, and issues of Cloud Computing Ecosystems, London: Springer-Verlag, 2010.

[13] M. Creeger, "CTO roundtable: cloud computing," in Communications of the ACM, 2009.

[14] N. Phaphoon, N. Oza, X. Wang and P. Abrahamsson, "Does Cloud Computing deliver the promised benefits for IT industry?," Proceedings of the WICSA/ECA, pp. 45-52, 2012.

[15] B. Grobauer, T. Walloschek and E. Stocker, "Understanding Cloud Computing Vulnerabilities," IEEE Security and Privacy, vol 9, no.2, pp. 50-57, 2011.

[16] L. Bautista, A. Abran et A. April, Design of a Performance Measurement Framework for Cloud Computing, Journal of Software Engineering and Applications, Vol. 5 No. 2, 2012, pp. 69-75, 2012.

[17] Omniti, "Reconnoiter Fault Detection and Trending," 2013.[Online]. Available: https://labs.omniti.com/labs/reconnoiter. [Accessed 05/ 2014].

[18] Cacti, "Cacti RRDTool," 2013.[Online]. Available: http://www.cacti.net. [Accessed 05/ 2014].

[19] ISO/IEC, ISO/IEC 15939:2007, Systems and Software Engineering, Measurement process, 2007.

[20] JCGM, International Vocabulary of Metrology – Basic and General Concepts and Associated terms, 2006.

[21] ISO/IEC, ISO/IEC 25010:2010 SOFTWARE ENGINEERING - Software Product Quality Requirements and Evaluation (Square) – System and Software Quality Models, ISO/IEC JTC 1/SC 7, 2010.

[22] ISO/IEC, ISO/IEC 25021:2012 Software Engineering -- Software product Quality Requirements and Evaluation (SQuaRE – Quality Measure Elements, ISO/IEC JTC 1/SC 7. 2012.

[23] ISO/IEC, ISO/IEC. ISO/IEC JTC 1 SC38:Cloud Computing Overview and Vocabulary, Geneva, Switzerland: International Organization for Standardization. 2012,

[24] J. Desharnais and A. Abran, "Software Measurement Methods: An analysis of Two Designs," Journal of Software Engineering and Applications, pp. 797-809, 2010.

[25] A. Alinezhad, A. Masaeli and N. M. M. Esfandiari, "Evaluation of Effectiveness of Implementing Quality Management System (ISO9001:2000) Using BSC Approach in NIGC", Journal of Industrial Engineering 6, 33-42, 2010.

[26] N. Mirzaei, cloud computing, Institute Report, Community Grids Lab, Indiana.Edu, 2008.

[27] NIST - National Institute of Standards and Technology, "The NIST Definition of Cloud Computing," CSD - TIL - NIST, Gaithersburg, MD, 2011.

[28] K.R Jackson; L. Ramakrishnan; K. Muriki; S. Canon,; Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud. IEEE, CloudCom 2010, pp 159-168.

[29] Microsoft – What is Microsoft Azure? – available on http://azure.microsoft.com/en-us/overview/what-is-azure/ [Accessed 02/2015]

[30] R. Teeter; K. Barksdale. Google Apps For Dummies. pp. 3–27. 2010

[31] Microsoft – Exchange 2010 system requirements – Available on https://technet.microsoft.com/en-us/library/aa996719(EXCHG.141).aspx [Accessed 12/2014]

 ISBN: 978-1-61208-388-9

Private Search Over Big Data Leveraging Distributed File System and Parallel Processing

Ayse Selcuk, Cengiz Orencik and Erkay Savas

Faculty of Engineering and Natural Sciences
Sabanci University, Istanbul, Turkey
Email:{ayseselcuk, cengizo, erkays}@sabanciuniv.edu

***Abstract*—In this work, we identify the security and privacy problems associated with a certain Big Data application, namely secure keyword-based search over encrypted cloud data and emphasize the actual challenges and technical difficulties in the Big Data setting. More specifically, we provide definitions from which privacy requirements can be derived. In addition, we adapt an existing work on privacy-preserving keyword-based search method to the Big Data setting, in which, not only data is huge but also changing and accumulating very fast. Our proposal is scalable in the sense that it can leverage distributed file systems and parallel programming techniques such as the Hadoop Distributed File System (HDFS) and the MapReduce programming model, to work with very large data sets. We also propose a lazy idf-updating method that can efficiently handle the relevancy scores of the documents in a dynamically changing, large data set. We empirically show the efficiency and accuracy of the method through an extensive set of experiments on real data.**

Keywords–Cloud computing; Big Data; Keyword Search; Privacy; Hadoop.

I. INTRODUCTION

With the widespread use of the Internet and wireless technologies in recent years, the sheer volume of data being generated keeps increasing exponentially resulting in a sea of information that has no end in sight. Although the Internet is considered as the main source of the data, a considerable amount of data is also generated by other sources such as smart phones, surveillance cameras or aircraft, and their increasing use in everyday life. Utilizing these information sources, organizations collect terabytes and even petabytes of new data on a daily bases. However, the collected data is useless unless it is possible to analyze and understand the corresponding information.

The emergence of massive data sets and their incessant expansion and proliferation led to the term, the *Big Data*. Accurate analysis and processing of Big Data, which bring about new technological challenges, as well as concerns in areas such as privacy and ethics, can provide exceptionally invaluable information to users, companies, institutions, and in general to public benefit. The information harvested from Big Data has tremendous importance since it provides benefits such as cost reduction, efficiency improvement, risk reduction, better health care, and better decision making process. The technical challenges and difficulties in effective and efficient analysis of massive amount of collected data call for new processing methods [1], [2], leveraging the emergent parallel processing hardware and software technologies.

Although the tremendous benefits of big data are enthusiastically welcomed, the privacy issues still remain as a major concern. Most of the works in the literature, unfortunately, prefer to disregard the privacy issues due to efficiency concerns since efficiency and privacy protection are usually regarded as conflicting goals. This is true to certain extent due to technical challenges, which, however, should not deter the research to reconcile them in a framework, that allows *efficient privacy-preserving process of Big Data*.

A fundamental operation in any data set is to find data items containing a certain piece of information, which is often manifested by a set of keywords in a query, namely *keyword based search*. An important requirement of an effective search method over Big Data is the capability of *sorting* the matching items according to their relevancy to the keywords in queries. An efficient ranking method is particularly important in Big Data setting, since the number of matching data items will also be huge, if not filtered depending on their relevance levels.

In this paper, we generalize the privacy-preserving search method proposed in our previous work [3] and apply it in the Big Data setting. The previous version [3], which is sequentially implemented, was only capable of working with small data sets that have sizes of only a few thousand documents. In order to get more prominent and explicit results using massive data, we leverage the Hadoop framework [4] which is based on the distributed file systems and parallel programming techniques. For relevancy ordering, we use the well known tf-idf weighting metric and adjust it to dynamic Big Data. Unlike the work in [3], we assume the data set is dynamic, which is an essential property of Big Data. Therefore, we propose a method that we call "Lazy idf Update" which approximates the relevancy scores using the existing information and only updates the inverse document frequency (idf) scores of documents when the change rate in the data set is beyond a threshold. Our analysis demonstrates that the proposed method is an efficient and highly scalable privacy preserving search method that takes advantage of the HDFS [4] and the MapReduce programming paradigm.

The rest of this paper is organized as follows. In the next section (Section II), we briefly summarize the previous work in the literature. The properties of Big Data and the new technologies developed for the requirements of Big Data

 ISBN: 978-1-61208-388-9

are summarized in Section III. In Section IV, we formalize the information that we hide in the protocol. The details of distributed file systems and the Hadoop framework are given in Section V. Section VI briefly summarizes the underlying search method of Orencik et al.[3]. The novel idf updating method for adjusting the tf-idf scoring for dynamically changing data set is explained in Section VII. In Section VIII, we discuss the results of the several experiments we applied on multi-node Hadoop setting. Section IX is devoted for the final remarks and conclusion.

II. Related Work

There are a number of works dealing with search over encrypted cloud data but most are not suitable for the requirements of Big Data. Most of the recent works are based on bilinear pairing [5]–[7]. However, computation costs of pairing based solutions are prohibitively high both on the server and on the user side. Therefore, pairing based solutions are generally not practical for Big Data applications.

Other than the bilinear pairing based methods, there are a number of hashing based solutions. Wang et al. [8] proposed a multi-keyword search scheme, which is secure under the random oracle model. This method uses a hash function to map keywords into a fixed length binary array. Cao et al. [9] proposed another multi-keyword search scheme that encodes the searchable database index into two binary matrices and uses inner product similarity during matching. This method is inefficient due to huge matrix operations and it is not suitable for ranking. Recently, Orencik et al. [3] proposed another efficient multi-keyword secure search method with ranking capability.

The requirements of processing Big Data led the big companies like Microsoft and Amazon to develop new technologies that can store and analyze large amounts of structured or unstructured data as distributed and parallel. Some of the most popular examples of these technologies are the Apache Hadoop project [4], Microsoft Azure [10] and Amazon Elastic Compute Cloud web service [11].

III. Challenges

As the name implies, the concept of Big Data is a massive dynamic set that contains a great variety of data types. There are several dimensions in Big Data that makes management a very challenging issue. The primary aspects of Big Data is best defined by its volume (amount of data), velocity (data change rate) and variety (range of data types) [12].

Unfortunately, standard off-the-shelf data mining and database management tools cannot capture or process these massive unstructured data sets within a tolerable elapsed time [13]. This led to the development of some new technologies for the requirements of Big Data.

In order to meet the scalability and reliability requirements, a new class of NoSQL based data storage technology referred as *Key-Value Store* [14]was developed and widely adopted.

This system utilizes associative arrays to store the key-value pairs on a distributed system. A key-value pair consists of a value and an index key that uniquely identifies that value. This allows distributing data and query load over many servers independently, thus achieving scalability. We also adapt the *key-value store* approach in the proposed method, where the details are explained in Section V.

IV. Privacy Requirements

In the literature, the privacy of the data analyzed by Big Data technologies is usually protected by anonymizing the data [15]. However, anonymization techniques are not sufficient to protect the data privacy. Although a number of searchable encryption and secure keyword search methods are proposed for the cloud data setting [5], [6], [9], none of them is suitable for Big Data.

A secure search method over Big Data should provide the following privacy requirements.

Definition 1. ***Query Privacy:*** *A secure search protocol has query privacy, if for all polynomial time adversaries A that, given two different set of search terms F_0 and F_1 and a query Q_b generated from the set F_b, where $b \in_R \{0, 1\}$, the advantage of A in finding b is negligible.*

Intuitively, the query should not leak the information of the corresponding search terms.

Definition 2. ***Data Privacy:*** *A secure search protocol has data privacy, if the encrypted searchable data does not leak the content (i.e., features) of the documents.*

The search method we adapted [3] satisfies both privacy requirements. We do not repeat the proofs here and refer to the original work.

V. Distributed File Systems

It is not possible to process large amounts of data that are in the order of terabytes by using only a single server, due to the storage and computation power requirements. Therefore, we utilize the cloud computing services by software as a service (SaaS), which provide the use of shared computing hardware resources over a network on a pay-as-you-go basis [16]. Most of the cloud computing platforms use the Hadoop [4], which is an open-source distributed and paralleled framework. It provides easy and cost-effective processing solutions for vast amounts of data. The Hadoop framework is comprised of two main modules, which are the HDFS [17] for storing large amounts of data and accessing with high throughput and the MapReduce framework for distributed processing of large-scale data on commodity machines.

A. Hadoop HDFS

The HDFS is an open source file system that is inspired by the Google file system (GFS) [18]. The HDFS architecture runs on distributed clusters to manage massive data sets. It is a highly fault-tolerant system that can work on low-cost hardware. In addition, the HDFS enables high throughput access for application data and streaming access for file system data. The HDFS is based on a master/slave communication model that is composed of a single master node and multiple data (i.e. slave) nodes. There exists a unique node called the NameNode that runs on the master node. The master node manages the file system namespace to arrange the mapping

 ISBN: 978-1-61208-388-9

between the files and the blocks and regulates the client access to the files [1].

B. Hadoop Mapreduce

Hadoop's MapReduce is based on Google's MapReduce algorithm [19]. The MapReduce programming model is derived from the Map and the Reduce functions which are used in functional programming beforehand. The MapReduce Programming model which processes massive data, provides large-scale computations for large clusters by dividing them into independent splits. The input data for the MapReduce is stored in the HDFS. The MapReduce utilizes the key-value pairs for distributing the input data to all the nodes in the cluster, in a parallel manner [20].

VI. Secure Search Method

The utilized privacy preserving search method is based on our previous work [3]. In this section, we briefly explain the method for completeness and refer the reader to [3] for the details. The search method is based on the minhashing technique [21]. Each document is represented by a constant length set called signature. During the similarity comparison, only the signatures are used and the underlying document feature sets are not revealed to the cloud. While this method cannot provide the exact similarity value, it can still provide a very accurate estimation. The signature of a document is defined as follows.

Definition 3. ***Minhash:*** *Let Δ be a finite set of elements, P be a permutation on Δ and $P[i]$ be the i^{th} element in the permutation P.* Minhash *of a set $D \subseteq \Delta$ under permutation P is defined as:*

$$h_P(D) = min(\{i \mid 1 \leq i \leq |\Delta| \ \wedge \ P[i] \in D\}). \quad (1)$$

For the signatures, λ different random permutations on Δ are used so the final signature of a document feature set D is:

$$Sig(D) = \{h_{P_1}(D), \ldots, h_{P_\lambda}(D)\}, \quad (2)$$

where h_{P_j} is the *minhash* function under permutation P_j.

A. Index Generation

The index generation is an offline operation initiated by the data owner and creates the secure searchable index that is outsourced to the cloud. The searchable index generation process is based on the bucketization technique [22], [23], which is a well known method for data partitioning.

For each minhash function and corresponding output pair, a bucket is created with bucket identifier B_k^i (i.e., i^{th} minhash function produces output k). Each document identifier is distributed to λ different buckets according to the λ elements of the corresponding signature. In addition to the document identifiers, the corresponding relevancy scores (i.e., tf-idf value) are also added to the bucket content ($V_{B_k^i}$).

Note that, both the bucket identifiers (B_k^i) and the content vectors ($V_{B_k^i}$) are sensitive information that needs to be encrypted before outsourcing to the cloud. The secure searchable index $\mathcal{I}$ is the combination of the encrypted bucket identifiers and the corresponding encrypted content vectors.

B. Query Generation and Search

The query is generated in the same way as generating secure index entries. Given the set of keywords to be searched for, the query signature is generated by using the same minhash functions used in the index generation phase. The elements of the query signature are indeed the identifiers of the buckets that include the documents that contain the queried keywords. The bucket identifiers in the query signature are encrypted using the same secret keys used in index generation. The query is this set of encrypted bucket identifiers. Independent of the number of queried keywords, the query signature, hence the query itself has constant length, which is λ.

In the search phase, the cloud server receives the query and sends the requested encrypted content vectors to the user. The user then decrypts the vectors and ranks the document identifiers according to their relevancy with the query using the tf-idf scores. Finally, the user retrieves the encrypted documents with the highest relevancy scores from the server. Alternatively, the operation performed by the user can be handled by a trusted proxy, relieving the burden on the user.

VII. Relevancy Scoring

In the information retrieval setting, the results are required to be ordered according to their relevancy with the query. A commonly used scoring metric for information retrieval is the tf-idf weighting [24]. Intuitively, it measures the importance of a term within a document for a database collection. The tf-idf weighting uses both the term frequency (tf) and the inverse document frequency (idf) metrics. The term frequency of a term w is the the normalized number of times that w occurs in a document. The inverse document frequency measures the rarity of a term within the whole data set. The tf-idf of a term w in a document D is calculated as given in (3).

$$\text{tf-idf}_{w,D} = \text{tf}_{w,D} \times \text{idf}_w. \quad (3)$$

Generally, some tools are used for calculating the tf-idf weight such as the Rapid Miner [25], which is a popular text mining tool. Li and Guoyong [26] proposed an efficient method for calculating the tf-idf algorithm based on the Hadoop Framework. We use this algorithm to calculate the tf-idf weights in our test data sets.

A. Lazy idf Update

The Big Data necessitates high velocity in the data set which means new data is added continuously. The tf-idf metric uses the inverse document frequency (idf) (i.e., rarity within the data set) of each keyword. Let a document D contain k previously indexed terms. As this new document D is included to the data set, the scores of all the documents that contain any of those k terms should be updated since their idf values change. However, dynamically applying this change for each data item, added or removed from the data set, is not feasible. Hence, we propose a *lazy idf updating* method which aims to maintain the scores of existing documents as they are and only set a new score for the newly added items. Moreover, calculating the idf of each term of a newly added data item is still a costly operation that requires scanning the whole data set. In order to reduce the cost of scoring, we propose keeping the idf values of the terms separately. As new data elements are

 ISBN: 978-1-61208-388-9

added, the idf values slightly change and the stored idf values will not exactly be correct. However, they still provide accurate estimates since the size of the existing data set is much larger than the size of the data elements added. In a timely bases (e.g., every 20 minutes), the whole data set is scanned and all the idf values are updated with the exact results.

Due to the privacy requirements, the server cannot see the actual documents but only stores the encrypted versions. It is not possible to calculate, neither the term frequencies, nor the inverse document frequencies from the encrypted data, therefore a trusted proxy should be used for updating the relevancy scores. Each new data item is first indexed and encrypted by the proxy and then uploaded to the server. Similarly, the idf value update operation is also done by the proxy. Therefore, the idf values that are separately stored are only kept in the trusted proxy. Since the idf update operation is performed by the proxy, the cloud server will be up and running during this period and the search operation can be done using the existing relevancy scores.

We assume that the size of the data set will be very large, hence the effect of the additional items on the idf values will be very limited. Note that, the term frequency (tf) part of the tf-idf score is calculated using only the document itself. Therefore, the change in the data set does not affect the tf values of the existing items. With this *lazy idf updating* method, very close estimates on the real tf-idf scores can be calculated in a very efficient way, hence it is suitable for the Big Data setting. The actual comparative results using a large, real data set is provided in Section VIII-C.

VIII. Experimental Results

In this section, we extensively analyze and demonstrate efficiency and effectiveness of the proposed method. The entire system is implemented by Java language using 64-bit Ubuntu 12.04 LTS operating system. In order to observe the benefits of distributed file systems, a multi-node Hadoop cluster is configured. The interface of Cloudera CDH4 with a three node (i.e., computer) cluster is utilized in the experiments. Two of the computers have an Intel Xeon CPU E5-1650 @ 3.5 GHz processor with 12 cores, 15.6 GB of main memory and the other computer has an Intel i7 @ 3.07 GHz processor with 8 cores and 15.7 GB of main memory.

In our experiments we used the Enron data set [27], which is a real data set that contains approximately $517,000$ email documents. Although the actual Enron data set is about 200 GB, we require a much smaller space as each document is represented by a single signature only, regardless of the size of the document.

A. Performance of the Method

In this section, we present the experiment results, where we measure the time spent for generating the secure searchable index and applying search operation.

The index generation time for $517,000$ documents, is given in Fig. 1 for different values of λ. The experiments demonstrate that the index generation, which is the most time consuming part of the method, can be done in only a few minutes. And the system can index about 2750 documents per second for $\lambda = 100$. Note that this operation is done only after the change in the data set, due to the documents added, exceeds a threshold. Moreover, since this operation is done by a trusted proxy, the cloud server can still continue to serve the incoming search requests, using the existing index.

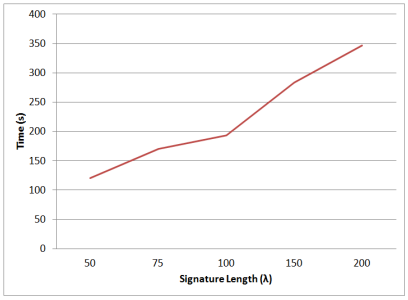


Figure 1. Index Generation Time as λ change

The search operation has two major parts. First the server fetches the content vectors of the queried buckets and sends them to the user. Then the user (or trusted proxy) decrypts those vectors and sorts the document identifiers according to the corresponding relevancy scores. Unfortunately, due to the distributed setting of the Hadoop file system, finding the queried buckets requires a search over all the created buckets. Fig. 2 demonstrates the average search time required both for the server and user sides, in the data set of size $517,000$ documents.

B. Accuracy of the Method

In the information retrieval community, two of the most common metrics for measuring the accuracy of a method are precision and recall. The metrics compare the expected and the actual results of the evaluated system. Precision measures the ratio of correctly found matches over the total number of returned matches. Similarly, recall measures the ratio of correctly found matches over the total number of expected results. Both precision and recall are real values between 0 and 1, where the higher the value the better the accuracy is.

In the case of a single term search, the proposed method guarantees all the matches that have non-zero relevancy scores, contain the searched term. Hence, retrieving all the items with non-zero scores satisfies perfect precision and recall. In the case of multiple keyword search, the matches with non-zero scores definitely contain at least one of the queried keywords

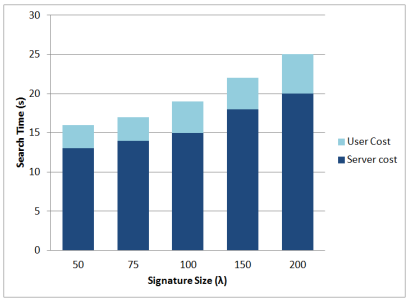


Figure 2. Search Time

but it may or may not contain all. We test the accuracy of the method for multi-term queries with $\lambda = 100$ (i.e., signature length) using the precision and recall metrics. The average precision and recall rates for a set of 20 queries with 2 and 3 keywords are given in Fig. 3 and 4, respectively. The retrieval ratio in the figures represents the ratio of the documents with nonzero scores that are considered as a relevant match with the query. The figures show that while the precision slightly decreases as retrieve ratio increases (i.e., more documents are considered as match), the recall increases. The retrieve ratio can be selected by the user according to the requirements of the application. The figures also show that the increase in the number of keywords decreases the precision but increases recall. The main reason of this is that, as the number of queried terms increases only very few documents contain all the queried terms which have a positive effect on recall. However, this also increases the documents with nonzero scores (i.e., contain at least one of the queried terms) which have a negative effect on precision.

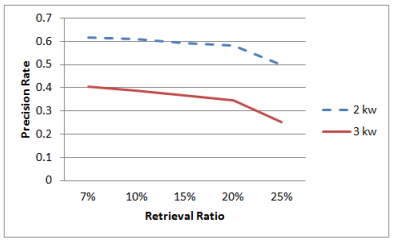


Figure 3. Average Precision Rate, $\lambda = 100$

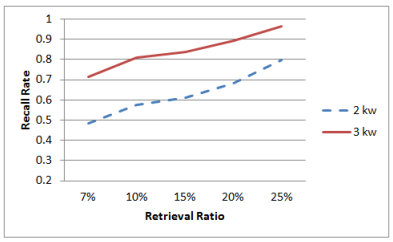


Figure 4. Average Recall Rate, $\lambda = 100$

Although the precision and recall metrics are very commonly used and very suitable for several problems such as conjunctive search and relational database search over structured data, they may not be very accurate for multi-keyword search over unstructured data. The main difference between search over structured and unstructured data is that, in the case of structured data, each field has an equal importance and the corresponding results should satisfy all the queried features. However, in the case of search over unstructured data, some of the queried features may be significantly more important than the others. For example, let a query has three features and a document contains only two of those features but with very high tf-idf scores. The precision and recall metrics will consider this document as a false match since it does not contain all the queried features, but in the case of Big Data we claim that this document is very relevant with the given query and should be considered as a match. It is important to note that, precision and recall metrics cannot consider the importance of the queried features in the compared document, hence may not perfectly measure the success rate of a search method over Big Data. Therefore, we also compare the output of the method with the ground truth. For calculating ground truth, the documents with top 50 scores in the data set are considered as the actual match results, where the complete tf-idf scores of the documents are used without any encryption. These actual results are then compared with the results evaluated by the system. The average precision and recall rates in comparison with the ground truth, for a set of 20 queries with 2 and 3 keywords and $\lambda = 100$, are given in Fig. 5 and 6, respectively. The figures show that, the actual accuracy of the method is quite promising when the tf-idf scores are considered in calculating the actual results, instead of the conjunctive (i.e., contain all terms) case.

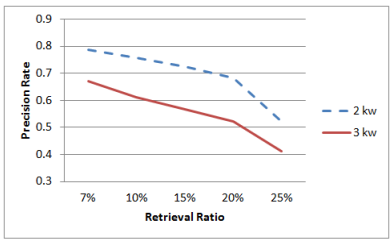


Figure 5. Average Precision Rate using Ground Truth, $\lambda = 100$

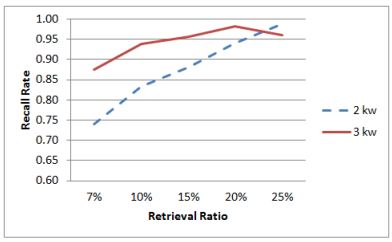


Figure 6. Average Recall Rate using Ground Truth, $\lambda = 100$

We also measure the effect of λ on accuracy. Fig. 7 and 8 show that an increase in λ has a positive effect on both precision and recall. However, increase in λ also linearly increases search and index generation times as shown in Section VIII-A and improvement in accuracy is very limited. Hence, an optimum value for λ should be set according to the properties of the data set used, which is set as 100 in our case.

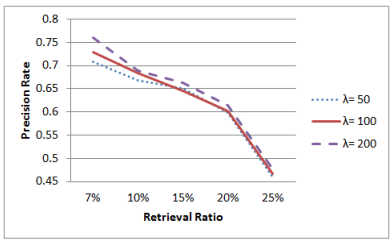


Figure 7. Average Precision Rate for different λ

C. Data Set Update

In Section VII-A, we propose a lazy update scheme that does not update the idf scores of the existing scores at each update but uses the existing scores as an approximation. In

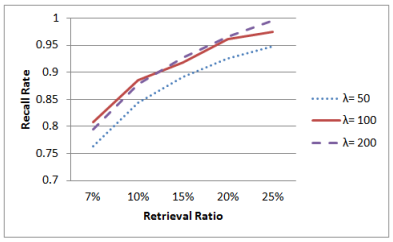


Figure 8. Average Recall Rate for different λ

this subsection, we provide the change rate of the idf due to update in the data set. We calculate the average idf scores of a data set of size $400,000$ documents while adding a new set of documents of size $10,000$. As Table I indicates, the effect of adding new documents is very low especially if the data set size is large, hence the lazy update does not reduce accuracy.

TABLE I. AVERAGE IDF VALUES

# documents	400,000	410,000	420,000	430,000
avg idf	2.26678	2.26716	2.26713	2.26717

Inserting index entries for documents added first requires calculating the corresponding signatures by a trusted proxy and than updating the encrypted bucket content vectors accordingly. We tested the update times for bulk insertions for 1000, 5000 and 10000 documents and the whole update operations are calculated as 52.5, 59.5 and 67 seconds, respectively. This shows that the update operation should be done for large sets of documents which is also suitable for the Big Data setting.

IX. Conclusions

In this work, we addressed the problem of applying an existing privacy-preserving search method for the case of Big Data. We utilized the search method of Orencik et al. [3] as the underlying search method and applied it for the HDFS and the MapReduce programming model. We implemented the entire system and tested for a three-node Hadoop with the Enron email data set and demonstrate the effectiveness and scalability of the system. We also proposed a *lazy idf update* method that can be used for dynamically changing large data sets and provide extensive results using a large real data set.

In the light of the promising results, we believe this method will increase the applicability of privacy preserving search over Big Data.

Acknowledgment

Ayse Selcuk was supported by ARGELA, under Grant Number 3014-07. Dr. Orencik and Dr. Savas were supported by TUBITAK under Grant Number 113E537.

References

[1] L. Wang, J. Tao, R. Ranjan, H. Marten, A. Streit, J. Chen, and D. Chen, "G-hadoop: Mapreduce across distributed data centers for data-intensive computing," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 739–750, 2013.

[2] L. Wang, M. Kunze, J. Tao, and G. von Laszewski, "Towards building a cloud for scientific applications," *Advances in Engineering Software*, vol. 42, no. 9, pp. 714–722, 2011.

[3] C. Orencik, M. Kantarcioglu, and E. Savas, "A practical and secure multi-keyword search method over encrypted cloud data," in *CLOUD 2013*, pp. 390–398, IEEE, 2013.

[4] http://hadoop.apache.org/core/, 2009. [accessed Nov 2014].

[5] B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search," *J. Netw. Comput. Appl.*, vol. 34, pp. 262–267, Jan. 2011.

[6] Z. Chen, C. Wu, D. Wang, and S. Li, "Conjunctive keywords searchable encryption with efficient pairing, constant ciphertext and short trapdoor," in *PAISI*, pp. 176–189, 2012.

[7] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Rou, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for boolean queries," in *Advances in Cryptology, CRYPTO 2013*, vol. 8042 of *Lecture Notes in Computer Science*, pp. 353–373, 2013.

[8] P. Wang, H. Wang, and J. Pieprzyk, "An efficient scheme of common secure indices for conjunctive keyword-based retrieval on encrypted data," in *Information Security Applications*, Lecture Notes in Computer Science, pp. 145–159, Springer, 2009.

[9] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," in *IEEE INFOCOM*, 2011.

[10] https://azure.microsoft.com/, 2014. [accessed Jan 2015].

[11] http://aws.amazon.com/ec2/, 2014. [accessed Jan 2015].

[12] D. Laney, "3d data management: Controling data volume, velocity and variety," 2001.

[13] C. Snijders, U. Matzat, and U. Reips, "Big data: Big gaps of knowledge in the field of internet," in *International Journal of Internet Science*, 2012.

[14] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels, "Dynamo: Amazon's highly available key-value store," in *Proceedings of 21st ACM SIGOPS Symposium on Operating Systems Principles*, SOSP '07, pp. 205–220, ACM, 2007.

[15] X. Zhang, L. T. Yang, C. Liu, and J. Chen, "A scalable two-phase top-down specialization approach for data anonymization using mapreduce on cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, pp. 363–373, Feb. 2014.

[16] Amazon Web Services, "What is cloud computing." http://aws.amazon.com/what-is-cloud-computing/. [accessed Dec 2014].

[17] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pp. 1–10, IEEE, 2010.

[18] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37, pp. 29–43, ACM, 2003.

[19] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, pp. 107–113, Jan. 2008.

[20] B. T. Rao and L. Reddy, "Survey on improved scheduling in hadoop mapreduce in cloud environments.," *International Journal of Computer Applications*, vol. 34, 2011.

[21] A. Rajaraman and D. Ullman, Jeffrey, *Mining of massive datasets*. Cambridge University Press, 2011.

[22] H. Hacigümüş, B. Iyer, C. Li, and S. Mehrotra, "Executing sql over encrypted data in the database-service-provider model," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, SIGMOD '02, pp. 216–227, ACM, 2002.

[23] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu, "Secure multidimensional range queries over outsourced data," *The VLDB Journal*, vol. 21, pp. 333–358, June 2012.

[24] H. S. Christopher D. Manning, Prabhakar Raghavan, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[25] https://rapidminer.com/. [accessed Jan 2015].

[26] B. Li and Y. Guoyong, "Improvement of tf-idf algorithm based on hadoop framework," 2012.

[27] "Enron email dataset." http://www.cs.cmu.edu/enron, Jan. 2012. [accessed Nov 2014].

 ISBN: 978-1-61208-388-9

A Conceptual Framework to Implement and Manage a Cloud Computing Environment

Gerard Conway, Marian Carcary, Eileen Doherty
Innovation Value Institute, Maynooth University
Maynooth, Ireland
emails: {gerard.conway, marian.carcary, eileen.doherty}@nuim.ie

***Abstract*-The proliferation of cloud adoption in recent years is driven by the potential for realizing benefits, such as reduced costs, improved agility and better resource utilization. However, there are many challenges to successfully delivering cloud-based services and these need to be understood and managed prior to cloud migration. Organizations need a systematic means of reviewing their business needs, so that the transition and subsequent management of a cloud environment is strategically planned. To address this requirement a Cloud Management Framework has been conceptualized that can be used for both the migration and the ongoing management of cloud-based services.**

Keywords-Cloud; Life cycle; Framework; Outsourcing; Cloud implementation.

I. INTRODUCTION

This paper describes a detailed organizational level conceptual framework for cloud implementation and management, developed through adopting a design science approach that involved collaboration with key academics and subject matter experts in the area of cloud technology.

A. An evolving landscape: Drivers and Barriers for Cloud adoption

In recent years, cloud computing adoption rates have proliferated, with the complete spectrum of businesses from large multinationals to smaller organizations migrating their Information Technology (IT) services to cloud platforms. There are multiple factors driving this organizational transition to a cloud environment, which include benefits such as cost reduction [1][2][6][8][18], increased scalability and agility [2][5][7][14][15][17], improved resource utilization [2][9][14][15], improved mobility and collaboration [1][10][14], and business continuity and disaster recovery capabilities [10]. However, despite this potential for benefits, it is not a panacea for all problems faced by organizations; in fact there are many challenges to successfully delivering cloud-based services [25]. Such challenges or barriers include security, and data protection [8][2][4][17], business continuity [2][10], statutory and legal requirements and restrictions on the flow of data across boundaries [8][16], lack of standardization and the resulting technology integration issues [10][17], and latency incurred in transferring data [2][18].

B. Approaches to cloud implementation

As cloud computing is a new and rapidly evolving area that presents many challenges, there are few detailed guidelines or best practices available to support an organization in its migration and ongoing management of a cloud environment. Various approaches to cloud adoption are highlighted in the literature, a number of which are discussed in this section.

Migration to the cloud computing environment reshapes a company's IT landscape. Hence, prior to transitioning to cloud, organizations need to weigh up the potential for benefit realization against the associated barriers and challenges. As such, organizations need a systematic means of reviewing their business needs, so that the transition to cloud computing is strategically planned and managed throughout the migration, and the associated implications are understood. Some prior studies have focused on such areas as cloud deployment and delivery models [3][11], and strategies for cloud adoption [4][8][12]. Others tend to deal with specific cloud issues such as security [22] and risk management [19], or are geared to specific areas such as Mobile clouds [20], are proprietary (e.g., Intel, Unisys, HP), or are based on existing frameworks such as the Information Technology Infrastructure Library (ITIL).

Previously, Conway and Curry proposed a life cycle approach to managing Cloud [4]. This combined Cullen's [26] outsourcing life cycle, with the Innovation Value Institute's (IVI) IT Capability Maturity Framework (IT-CMF). IT-CMF is an innovative and systematic framework, enabling Chief Information Officers (CIOs) and Chief Financial Officers (CEOs) to understand and improve their organization's maturity in order to enable optimal business value realization from IT investments [21]. The framework identifies 35 critical IT capabilities and defines maturity models across five levels for each capability. A core function of IT-CMF is to act as an assessment tool and a management framework.

The approach taken by Conway and Curry was to adapt Cullen's work on outsourcing to produce a life cycle for Cloud, and to map the critical capabilities, as defined by IT-CMF, to each lifecycle step - see Figure 1 and Tables 1, 2 and 3 for details.

This approach was used to develop an Executive (i.e., high-level) assessment framework to provide an organization with a management structure to assess the following:

 ISBN: 978-1-61208-388-9

- The readiness/maturity of an organization to move to cloud.
- The day-to-day management of an organization's cloud environment.
- The identification of services that can be provided by cloud.

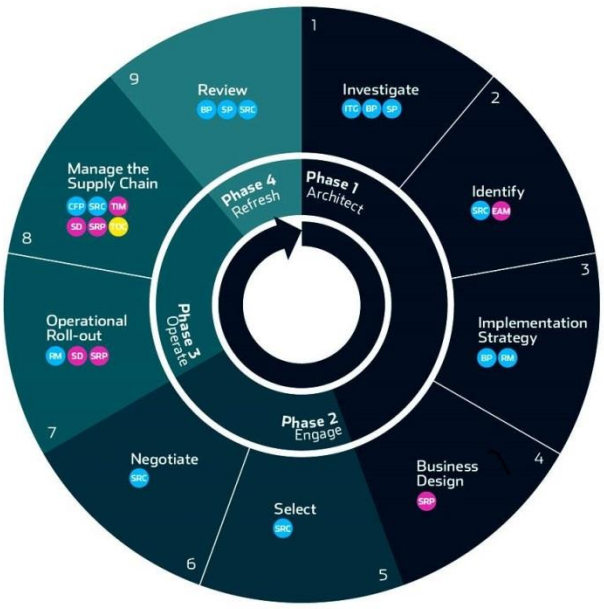


Figure 1. Cloud Executive Level Framework

TABLE 1. CLOUD LIFECYCLE PHASES (EXECUTIVE LEVEL)

Lifecycle Phase	Description
Architect	Investigate and plan the cloud project.
Engage	Select a service provider that can deliver the required cloud service.
Operate	Implement and manage the cloud service on a day-to-day basis.
Refresh	Review cloud services on an ongoing basis.

TABLE 2. CLOUD LIFECYCLE STEPS (EXECUTIVE LEVEL)

Lifecycle Phase	Description
Investigate	Provide insight and an understanding of what an organization wants to achieve by moving to the cloud, and what goals and expectations are to be met.
Identify	Objectively assess what areas of the business are appropriate to outsource to the cloud and what impact this will have on the current delivery model.
Implementation strategy	Define at a strategic level how the cloud services that are to be outsourced will be rolled out.
Business design	Design what is to be outsourced to the cloud and what the future state will look like.
Select	Based on the requirements and the other criteria defined by the Architect phase, select the best supplier based on value, sustainability, and quality.
Negotiate	Complete the final negotiation, pick the preferred supplier, get internal approval and sign the contract(s).
Operational roll out	Put together a project team that will manage the transition of the agreed services to the new cloud environment.
Manage Cloud Services	Manage cloud service(s) as efficiently and effectively as possible.
Review	Review cloud service requirements based on: the cloud service itself, other changes within the business, changes within the supplier organization, or the need to change the supplier.

TABLE 3. CLOUD LIFECYCLE CAPABILITIES (EXECUTIVE LEVEL)

Phase	Step	Capability
Architect	*Investigate*	IT Leadership and Governance (ITG)
		Business Planning (BP)
		Strategic Planning (SP)
	Identify	Sourcing (SRC)
		Enterprise Architecture Management (EAM)
	Implementation Strategy	Business Planning (BP)
		Risk Management (RM)
	Design	Service Provisioning (SRP)
Engage	*Select*	Sourcing (SRC)
	Negotiate	Sourcing (SRC)
Operate	*Roll-out*	Risk Management (RM)
		Solution Delivery (SD)
		Service Provisioning (SRP)
	Manage the Supply Chain	Capacity Forecasting and Planning (CFP)
		Sourcing (SRC)
		Technical Infrastructure Management (TIM)
		Solution Delivery (SD)
		Service Provisioning (SRP)
		Total Cost of Ownership (TCO)
Refresh	*Review*	Business Planning (BP)
		Strategic Planning (SP)
		Sourcing (SRC)

C. Requirement for a Detailed Micro-Level Organizational Framework for Cloud Implementation and Management

A review of the literature illustrates there to be a number of approaches to cloud migration available, however many

serve to provide a high-level structure [18][20], as opposed to a more micro-level of prescription or guidance that can be executed at a practitioner level. In effect, what is missing is an independent holistic management framework that allows an organization to consider all aspects of cloud from a technical, people and process perspective.

Conway and Curry's original Executive Level Management Framework [4] provides a potential solution, however although organizations have used this framework successfully, they observe that significantly greater detail is required, to adequately address the problems posed by cloud. This paper attempts to address this requirement through the development of a framework at a detailed micro-level to support the migration to and management of a cloud environment.

The structure of the paper is as follows; Section 1 introduced and provided justification for the development of a cloud framework at the micro-level of prescription; Section 2 outlines the methodological research approach adopted; Section 3 presents the detailed micro-level cloud implementation and management framework; Section 4 discusses and provides an overview of the key conclusions.

II. RESEARCH METHODOLOGY

Firstly, and closely mirroring the approach taken by Conway and Curry [4], the IT-CMF was mapped to the Cloud Lifecycle to establish the key capabilities an organization should consider in order to successfully migrate to, and manage a cloud environment (Figure 2). See [27] for a detailed explanation of the IT-CMF capabilities.

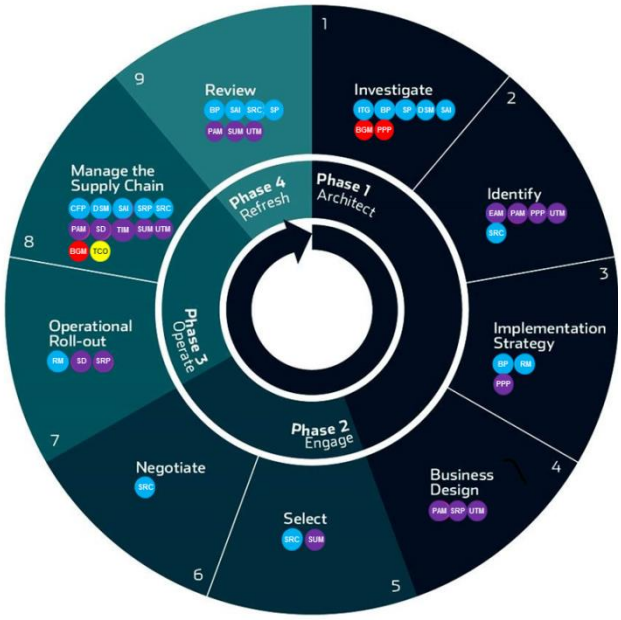


Figure 2. Detailed micro-level cloud framework

Secondly, a qualitative approach was taken to the framework development process (using both a workgroup and semi-structured interviews with key stakeholders). The workshop/workgroup approach was undertaken in conjunction with the Innovation Value Institute (IVI) consortium comprising of leading organizations from industry (including: Microsoft, Intel, SAP, Chevron, Cisco, The Boston Consulting Group, Ernst & Young, and Fujitsu) and academia. The consortium adopted an open innovation model of collaboration that engages academia and industry in scholarly work to amalgamate leading academic theory with corporate thought leadership in order to advance practices for managing IT for business value and innovation. This workgroup followed a design process with defined review stages and development activities that were based on Design Science Research (DSR) guidelines [23]. A cloud workgroup of ten subject matter experts, comprising researchers, industry practitioners and academics drawn from this consortium was established to develop the framework for systematically managing cloud projects. In addition to incorporating the insights of workgroup members throughout the framework development process, a qualitative approach to empirical data collection was adopted. Semi-structured interviews with cloud stakeholders across 11 organizations were conducted in order to capture the views of key domain experts and to understand current practice and barriers to managing cloud projects. These included organizations that had both successfully delivered, and that had failed to deliver, cloud-based projects. The interview approach enabled depth, nuance and complexity to be captured [24], and the insights gathered were used to inform and revise the framework's development.

III. DEVELOPMENT OF A DETAILED MICRO LEVEL FRAMEWORK TO IMPLEMENT AND MANAGE A CLOUD COMPUTING ENVIRONMENT

In the original Executive level framework, Conway and Curry [4] identified 11 critical capabilities (Table 3) to provide an executive high-level maturity assessment comprising 24 questions, so that organizations could rapidly understand their ability to migrate and manage their cloud environment. Based on the feedback from organizations that used the Executive level framework, a new detailed micro-level framework was developed as shown in Figure 2 and Table 4. This micro-level framework provides organizations with an in-depth maturity assessment based on an expanded list of 18 critical capabilities (Table 4) that encompasses 169 maturity questions.

 ISBN: 978-1-61208-388-9

TABLE 4. CLOUD LIFECYCLE CAPABILITIES (DETAILED LEVEL)

Phase	***Step***	***Capability***
Architect	*Investigate*	IT Leadership and Governance (ITG)
		Budget Management (BGM)
		Business Planning (BP)
		Demand and Supply Management (DSM)
		Portfolio Planning and Prioritization (PPP)
		Service Analytics and Intelligence (SAI)
		Strategic Planning (SP)
	Identify	Enterprise Architecture Management (EAM)
		People Asset Management (PAM)
		Portfolio Planning and Prioritization (PPP)
		Sourcing (SRC)
		User Training Management (UTM)
	Implementation Strategy	Business Planning (BP)
		Portfolio Planning and Prioritization (PPP)
		Risk Management (RM)
	Design	People Asset Management (PAM)
		Service Provisioning (SRP)
		User Training Management (UTM)
Engage	*Select*	Sourcing (SRC)
		Supplier Management (SUM)
	Negotiate	Sourcing (SRC)
Operate	*Roll-out*	Risk Management (RM)
		Service Provisioning (SRP)
		Solution Delivery (SD)
Operate	*Manage the Supply Chain*	Budget Management (BGM)
		Capacity Forecasting and Planning (CFP)
		Demand and Supply Management (DSM)
		People Asset Management (PAM)
		Service Analytics and Intelligence (SAI)
		Service Provisioning (SRP)
		Solution Delivery (SD)
		Sourcing (SRC)
		Supplier Management (SUM)
		Technical Infrastructure Management (TIM)
		Total Cost of Ownership (TCO)
		User Training Management (UTM)
Refresh	*Review*	Business Planning (BP)
		Service Analytics and Intelligence (SAI)
		Sourcing (SRC)
		Strategic Planning (SP)

The micro-level framework is designed for three different use case scenarios as follows:

- A complete assessment using all 18 capabilities.
- An assessment based on specific phases or steps of the cloud life cycle: e.g., limit the assessment to the negotiate step (where the preferred service providers are selected and contracts finalized).
- An assessment based on one specific cloud issue: e.g., security.

Each of these use case scenarios provides the following:

- It supports an organization in understanding its current maturity using IVI's maturity model.
- It identifies an organization's future target maturity level.
- It determines the importance the organization places on each capability in the context of the cloud lifecycle.

The output from an assessment highlights areas of low maturity, and the gap between current and target maturity relative to its importance to the organization's needs. This supports prioritizing those capabilities that will benefit from short, medium, and long-term development. The detailed micro-level assessment also provides an organization with an improvement roadmap, detailing the actions necessary to drive improvement across the areas under investigation. This is achieved by providing guidance on the practices typical at each maturity level, the value-oriented outcomes resulting from effective implementation of these practices, and a series of metrics to monitor progress and performance of these practices over time.

IV. DISCUSSION AND CONCLUSION

As highlighted in the literature, there are various cloud approaches that can be utilized in the organizational context; however it is acknowledged that these frameworks do not offer a holistic approach to managing the transition to or management of the cloud environment. Consequently, this paper presents a detailed micro-level conceptual framework developed by key academics and industry practitioners using a Design Science approach to address this gap. Engagement with these subject matter experts was key in the framework's development and refinement. For example, one key insight was that the framework needed to provide organizations with the flexibility to use it where and when it was needed, and to allow it to be tailored to an organization's specific requirements. This was incorporated into the framework by:

- Allowing it to be consumed in its entirety or tailored for use at a particular stage, e.g., the Identify stage.
- Linking each capability to a specific cloud issue thereby allowing an organization to tailor the framework to address a specific area, e.g., security.

This framework provides organizations with a management tool to enable them to better understand their enterprise IT capability maturity to position, evaluate, introduce and manage cloud services, and their associated strengths and weaknesses in these areas. The framework further provides organizations with practical improvement roadmaps for cloud migration and management which are grounded in industry best practice at a detailed level of prescription. Whilst the framework does not claim to address

 ISBN: 978-1-61208-388-9

all of the issues faced by an organization when implementing and managing cloud, it is proposed as a good starting point in the transition and subsequent management of the cloud environment. Limitations of the study are its small sample and its focus on large multinational organizations. Future studies could address these limitations by increasing the sample size or by undertaking further research with Small and Medium-sized Enterprises (SMEs). The study and the resultant conceptual framework adds value to academics, as it provides new insights into an area where there is currently a research deficit, specifically in the transition to a cloud computing environment. This paper may assist policy makers in this area through highlighting the key challenges facing firms in cloud adoption. Finally, from an industry perspective it is hoped that researchers in this area may utilize this framework and provide further validation and refinement through its use.

ACKNOLEGEMENT

The work presented in this paper has been funded by Enterprise Ireland under Grant CC/2009/0801.

REFERENCES

[1] A. Aljabre, "Cloud computing for increased business value", International Journal of Business and Social Science, vol. 3, no. 1, 2012, pp. 234-239.

[2] M. Armbrust, et. al., "A view of cloud computing", Communications of the ACM, vol. 53, no. 4, 2010, pp. 50-58.

[3] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: vision, hype, and reality for delivering IT services as computing utilities", Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications, 2008, pp. 5-13.

[4] G. Conway and E. Curry, "Managing cloud computing – a lifecycle approach". Proceedings of the 2nd International Conference on Cloud Computing and Services Science. April 18-21, 2012, pp 198-207, Porto, Portugal.

[5] EMCC, "Knowledge-intensive business services: trends and scenarios", 2012, Online at: http://www.eurofound.europa.eu/emcc/content/source/eu05014a.htm Retrieved February 2015.

[6] P. Geczy, N. Izumi, and K. Hasida, "Cloudsourcing: managing cloud adoption. Global Journal of Business Research", 6(2), 2012, pp.57-70.

[7] A. Goscinski and M. Brock, "Toward dynamic and attribute based publication, discovery and selection for cloud computing", Future Generation Computer Systems, 26, 2010, pp.947-70.

[8] B. Iyer and J. C. Henderson, "Preparing for the future: understanding the seven capabilities of cloud computing",MIS Quarterly Executive, 9(2), 2010, pp.117-131.

[9] V. Kundra, "Federal cloud computing strategy", 2011, Online at: http://www.cio.gov/documents/federal-cloud-computing-strategy.pdf Retrieved February 2015.

[10] Kynetix Technology Group, "Cloud computing – a strategy guide for board level executives", 2009 Online at: http://dastikop.blogspot.ie/2012/08/cloud-computing-strategy-guide-for.html Retrieved February 2015.

[11] S. Leimeister, C. Riedl, M. Bohm, and H. Krcmar, "The business perspective of cloud computing: actors, roles and value networks". Proceedings of the 18th European Conference on Information Systems. 7th-9th June, 2010, Pretoria, South Africa.

[12] C. Loebbecke, B. Thomas, and T. Ulrich, "Assessing cloud readiness at Continental AG", MIS Quarterly Executive, 11(1), 2012, pp. 11-23.

[13] E. Luoma and T. Nyberg, "Four scenarios for adoption of cloud computing in China". Proceedings of the European Conference on Information Systems 2011, Online at http://aisel.aisnet.org/ecis2011/123 Retrieved February 2015.

[14] F. T. Neves, F.C. Marta, A. M. Correia, and M. Neto, "The adoption of cloud computing by SMEs: identifying and coping with external factors". Proceedings of the 11th Conference of the Portuguese Association of Information Systems. 19th-21st October, 2011, Lisbon, Portugal. Online at: http://run.unl.pt/bitstream/10362/6166/1/Neves_Marta_Correia_Neto_2011.pdf. Retrieved February 2015.

[15] J. Pyke, "Now is the time to take the cloud seriously", White Paper, 2009, Online at: www.cordys.com/cordyscms_sites/objects/bb1a0bd7f47b1c91ddf36ba7db88241d/time_to_take_the_cloud_seroiusly_online_1_.pdf. Retrieved February 2015.

[16] SIM Advanced Practices Council "Wisdom of clouds: learning from users", 2011,Online at: https://simnet.site-ym.com/store/default.asp, Retrieved February 2015.

[17] N. Su, "Emergence of cloud computing: an institutional innovation perspective". Proceedings of the 32nd International Conference on Information Systems, , 2011, pp. 257-259.

[18] H. Yang and M. Tate, "Where are we at with cloud computing?: a descriptive literature review". Proceedings of the 20th Australasian Conference on Information Systems, 2nd-4th December, 2009, pp. 2-4.

[19] C. Horwath, W. Chan, E. Leung, and H. Pili, "Enterprise risk management for Cloud Computing", 2012, Online at: http://www.coso.org/documents/Cloud%20Computing%20Thought%20Paper.pdf, Retrieved February 2015.

[20] J.S. Mohammad, "Mobile Cloud Management: A New Framework", 23rd Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2012, pp. 44-53.

[21] M. Curley, "Managing Information Technology for Business Value", 2004, Intel Press.

[22] M. Almorsy, J. Grundy and A. S. Ibrahim, "Collaboration-based cloud computing security management framework". 2011 IEEE International Conference on Cloud Computing. Online at: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6008653#, Retrieved February 2015, pp. 364-371.

[23] A. R. Hevner, S. R. March, J. Park, and S. Ram, "Design Science in Information Systems Research", MIS Quarterly, 28, (1), 2004, pp. 75-105.

[24] J. Mason, Qualitative Researching, 2002, London, Sage Press.

[25] C. Brooks. "Heroku learns the hard way from Amazon EC2 outage", 2010. Online at: http://searchcloudcomputing.techtarget.com/news/1378426/Heroku-learns-from-Amazon-EC2-outage, retrieved Feb 2015.

[26] S. Cullen, P. Seddon, and L. Wilcox, "Managing Outsourcing, The Life Cycle Imperative", MIS Quarterly Executive, vol. 4, no.1, 2005, pp. 229-256.

[27] M. Carcary and O. Zlydareva, "Investigating the application of the IT-CMF in maturing strategic business-IT alignment" in Proceedings of the 8th European Conference on Information Management and Evaluation, Ghent, Belgium, ECIME 2014, pp. 29-33.

 ISBN: 978-1-61208-388-9

A Reliability Assessment Framework for Cloud Applications

Xiaowei Wang
Institute of Computer Science
University of Goettingen
Goettingen, Germany
xiaowei.wang@cs.uni-goettingen.de

Jens Grabowski
Institute of Computer Science
University of Goettingen
Goettingen, Germany
grabowski@cs.uni-goettingen.de

***Abstract*—Cloud computing enables users to use computing resources, platforms and applications with reduced deployment and maintenance cost. The reliability of cloud applications becomes one of the key concerns of cloud service providers and users. Meanwhile, the deep dependency stack of layered cloud objects makes it challenging to evaluate the reliability of cloud applications. To tackle this problem, we propose a layered dependency graph-based reliability assessment framework. To verify our framework, we conduct an initial case study which shows its feasibility.**

Keywords-cloud application; reliability assessment; deep dependency.

I. INTRODUCTION

Cloud computing emerges as a promising paradigm that has potential to provide computing services as a utility [1]. It virtualizes computing resources (such as servers, networks, platforms and software) into resource pools, which can be used on demand via the Internet. In recent years, increasingly more companies and organizations have migrated their applications and data into clouds to reduce the in-house hardware and maintenance cost. Beside the rapid growth of cloud computing, the reliability of cloud applications is still on the road to satisfy cloud users. As unreliable cloud services may lead to revenue loss and data loss, the assessment and improvement of cloud system and applications' reliability attract significant attention of both academia and industry [2][3].

However, the deep dependency stack [4] of cloud objects, such as physical servers, virtual machines (VMs), platforms, services and management software etc., in different layers: Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS) and physical infrastructure, makes it a system-level task to assess the reliability of cloud applications, since the reliability of objects in upper layers is dependent on the reliability of objects in lower layers. The hierarchical dependency among cloud objects makes it tough to find out root causes of failures, i.e., where to put efforts to improve the reliability.

To address this issue, we propose a framework to assess and analyze the reliability of cloud applications. The framework utilizes a layered dependency graph to model dependencies between related cloud objects and the application deployed on clouds. Furthermore, a reliability assessment method is proposed based on the layered dependency graph. According to the modeled dependency and the field monitoring data, the reliability of each object and the application is assessed.

The rest of the paper is organized as follows: Section II discusses related work. Section III describes the layered dependency graph. Section IV illustrates the reliability assessment method of cloud objects. Section V introduces our framework. Section VI shows a preliminary case study of our framework. Section VII presents the conclusion and future work.

II. RELATED WORK

Recently, the assessment of cloud applications' reliability has become a hot research field. Zheng et al. [5] propose a framework to select the most significant components to determine the optimal reliability strategy for component-based cloud applications. But [5] does not take hardware components into consideration. In [6], Dai et al. divide the cloud service failures into request stage failures and execution stage failures, and then employ Markov model and graph theory to model and analyze the reliability of cloud services. Thanakornworakij et al. [7] propose a reliability model for high performance computing applications considering the correlation of software failures and hardware failures. However, neither [6] nor [7] considers the structure of the application. In [8], Tamura et al. propose a reliability model for open source cloud software focusing on the operational environment fluctuation. But [8] is more about the reliability of cloud systems rather than the reliability of cloud applications. Comparing with existing work, the framework proposed in this paper is capable to assess the reliability of cloud applications combining the reliability of software as well as hardware objects based on the structure of the application and the deployment of service instances.

III. LAYERED DEPENDENCY GRAPH

A cloud application is composed of several services, each of which has one or more service instances. For simplicity, we assume that one physical server can host more than one VM, but one VM can hold only one service instance. We define the chain of dependencies among service instances, VMs and physical servers as deep dependencies [4].

As assumption, some or all VMs used by a service may be deployed on the same physical server, as Figure 1 shows, which means that the failure of one physical server may bring down several service instances. The above case should be avoided when we improve the reliability of cloud applications. Therefore, a Layered Dependency Graph (LDG) is employed to model deep dependencies. A LDG contains three layers from bottom to top: physical server layer, VM layer and service instance layer, as shown in

 ISBN: 978-1-61208-388-9

Figure 1. Cloud objects that are taken into consideration are service instances, VM instances and physical servers. In the service instance layer, service instances of one service type are clustered.

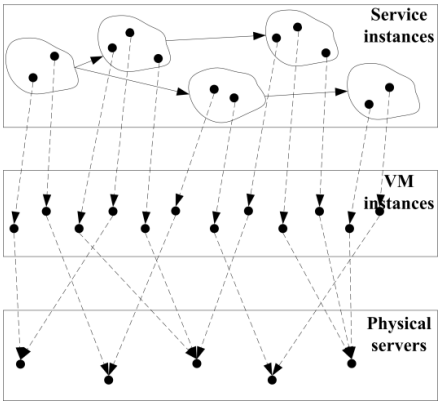


Figure 1. An example of the layered dependency graph.

We define two kinds of dependencies between cloud objects. The *function dependency* (solid arrows in Figure 1) is the relationship between two services that a service needs another one for its full function, e.g., a website needs a database to store users' information. And the *deployment dependency* (dashed arrows in Figure 1) is the relationship between objects in adjacent layers indicating that an object is deployed onto another one, e.g., a service instance is deployed on a VM.

IV. RELIABILITY ASSESSMENT

We define reliability as "the ability of a system or component to perform its required functions under stated conditions for a specified period of time" [9]. To illustrate the impact of dependencies to objects' reliability, we assume that the reliability (R) of a cloud object is determined by the reliability of itself (inner reliability, denoted by r) and the reliability (R_i) of objects on which it depends [10]. Based on the LDG model, the reliability of an object is represented as

$$R = r * \prod_{i=1}^{n} R_i \tag{1}$$

where n is the number of dependent objects. Equation (1), with the given inner reliability of all objects and time, can be used to obtain the application reliability based on the LDG. The reliability of cloud objects will be calculated in the following sections.

A. Physical Server Reliability

Physical server reliability (R_{PS}) is defined as the probability that a physical server performs its functions without failures in a period of time. Physical servers are considered failed when they crash or are unreachable. Physical servers depend on no other objects, as a result, their reliability is fully determined by their inner reliability (r_{PS}). Because physical servers work with constant failure rates (λ_{PS}) during the operational phase [6], we utilize the exponential reliability model to assess the physical server reliability with

$$R_{PS} = r_{PS} = e^{-\lambda_{PS} t} \tag{2}$$

where t is the working time of the physical server. λ_{PS} is usually evaluated by mean time to failure (MTTF) with

$$\lambda_{PS} = \frac{1}{MTTF}. \tag{3}$$

B. VM and Service Instance Reliability

As discussed in [8], the exponential distribution performs well for modeling software failures. Thus, we estimate the inner reliability of VM instances and service instances with the exponential reliability model by

$$r = e^{-\lambda t} \tag{4}$$

where λ is the failure rate of the object and t is the running time of the object. Software failures are not able to be tolerated by redundancy, except for timing or transient failures (called Heisenbugs) [11], which are usually caused by the complicated runtime environment. We assume that not only service instances of a service but also VMs on a physical server have the same failure rate.

VMs are considered failed if not in the running state or not reachable. Failures of the network, hypervisors and the cloud manager etc. that may lead to VM failures are deemed failures of VMs. Therefore, a VM will only fail due to VM failures or failures of the corresponding physical server, which means that all VMs on the same physical server will run or fail simultaneously, since they run in the same environment. The inner reliability of every VM on a physical server (r_{VM}) is

$$r_{VM} = e^{-\lambda_{VM} t} \tag{5}$$

where λ_{VM} is the internal failure rate of the VM.

Combining with the reliability of physical servers, we get the reliability of a VM with

$$R_{VM} = r_{VM} * R_{PS} \tag{6}$$

Equations (5) and (6) also apply for the inner reliability (r_{SI}) and reliability (R_{SI}) of service instances, respectively, with the service instance internal failure rate λ_{SI}, by

$$r_{SI} = e^{-\lambda_{SI} t} \tag{7}$$

$$R_{SI} = r_{SI} * R_{VM}. \tag{8}$$

 ISBN: 978-1-61208-388-9

C. Service Reliability

A service fails when all of its instances fail or at least one service that it depends on fails. Heisenbugs are usually caused by the complicated runtime environment, so, the inner reliability of all instances of a service is the same. Therefore, the inner reliability of a service is determined not only by the number of instances but also by the diversity of its VMs (i.e., deep dependencies).

Considering a service with three instances (Figure 2) deployed on three VMs, we will discuss the method to assess the inner reliability of the service (r_s) in different scenarios.

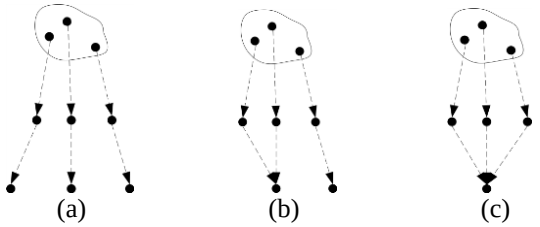


Figure 2. Scenarios of deploying a service with three instances.

a) If all three VMs are deployed on three different physical servers, as Figure 2(a) shows, then according to the assumption, the service will fail only when all the VMs fail (possibly caused by failures of all the physical servers), so, the service inner reliability is estimated with

$$r_s = r_{SI} * [1 - \prod_{i=1}^{3} (1 - R_{VM_i})] \quad (9)$$

where R_{VM_i} is the reliability of the *i*th VM.

b) If two of three VMs (e.g., VM_1 and VM_2) are deployed on the same physical server, because they run or fail simultaneously, it equals the scenario that the service has only two VMs deployed on two different physical servers and the service inner reliability is estimated with

$$r_s = r_{SI} * [1 - (1 - R_{VM_2})(1 - R_{VM_3})]. \quad (10)$$

c) If all three service instances are deployed on the same physical server, which means that any failures of a service instance, a VM or a physical server will make the service fail. It equals the scenario that only one service instance is running on this physical server. The service's inner reliability can be estimated with

$$r_s = r_{SI} * R_{VM}. \quad (11)$$

We can draw the conclusion from case *b)* and *c)* that if we want to improve the reliability of a service by increasing the amount of the service instance, the redundant services must be on different physical servers (regardless of performance issues).

Finally, the service reliability (*Rs*) is the multiplication of its inner reliability and the reliability of all services it depends on

$$R_s = r_s * \prod_{i=1}^{n} R_{s_i} \quad (12)$$

where R_{s_i} is the reliability of the *i*th dependent service and *n* is the number of dependent services.

D. Application Reliability

The application reliability is equal to the reliability of the service (e.g., S_1), which directly interacts with users and no other services are dependent on it, and is calculated with

$$R_{app} = R_{s_1}. \quad (13)$$

With the method of calculating the reliability of different kinds of cloud objects and the structure of the application, we can assess the reliability of cloud applications.

V. FRAMEWORK

In this section, we present a framework containing three components: a monitor, a dependency analyzer and a reliability analyzer, as Figure 3 shows.

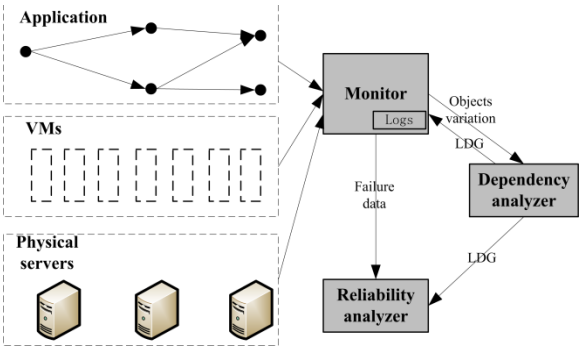


Figure 3. The reliability assessment framework.

A. Monitor

The responsibility of the monitor is twofold. The first task is to monitor and log the status, especially the failures, of all objects included in the LDG. The second task is to inform the dependency analyzer when any object fails, recovers from failures or joins the system. For instance, when a new service instance is started, the monitor transfers the name of the service to the dependency analyzer. The dependency analyzer will firstly query the information of the new service instance, the corresponding VM and physical server from the cloud manager, and secondly update and return the new LDG to the monitor.

B. Dependency Analyzer

The dependency analyzer is designed to create and update the LDG of the application. When deploying an application to the cloud, users need to input the initial

 ISBN: 978-1-61208-388-9

function dependencies between services to the dependency analyzer. When the application is deployed, the dependency analyzer gets deployment dependencies from the cloud manager to build the LDG.

C. Reliability Analyzer

The reliability analyzer is responsible for assessing the reliability of the application and each object in the LDG by field failure data of all objects obtained from the monitor and the dependencies obtained from the dependency analyzer.

VI. PRELIMINARY CASE STUDY

To evaluate the framework, we implement a prototype with Java based on a Cloudify [12] PaaS cloud which is built on top of a private OpenStack [13] IaaS cloud.

The dependency analyzer collects the dependency information from Cloudify and OpenStack to create LDGs. Cloudify is employed to monitor service instances and VMs, and Ganglia [14] is used to monitor physical servers.

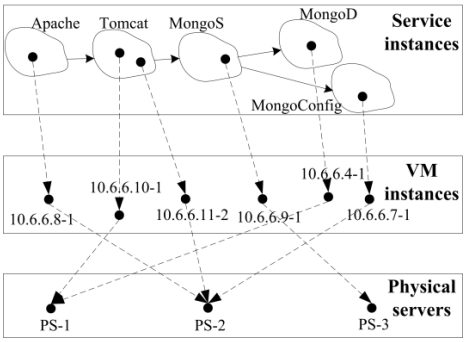


Figure 4. The created layered dependency graph.

We deploy a website using the Apache HTTP server [15] as the load balancer, Apache Tomcat [16] as the application server and MongoDB [17] (including three kinds of services, Mongos, MongoConfig and MongoD) as the database. The load balancer interacts directly with users and depends on Tomcat to fulfill functions. Tomcat is dependent on MongoS which depends on MongoConfig and MongoD. The LDG created by the dependency analyzer is shown in Figure 4.

We monitored the website and the cloud system for three days and obtained usage information of 71 visitors with 905 hits. From the monitoring logs, the internal failure rates of Apache HTTP server and the website deployed on Tomcat are 11/72 per hour and 4/72 per hour respectively. Based on the proposed reliability assessment method, the inner reliability of Apache HTTP server and the website is 0.8583 and 0.9460 respectively. So, the reliability of Apache HTTP server in one hour is

$$R_{apache} = 0.8583 * 0.9460 = 0.8120 .$$

No VM failures or physical server failures are observed during the three days, so, according to the proposed reliability assessment method, the application reliability is also 0.8120 in one hour.

VII. CONCLUSION AND FUTURE WORK

In this paper, we propose a framework for assessing the reliability of cloud applications based on LDGs. As the preliminary case study shows, the framework can assess the reliability of cloud objects and applications. However, the preliminary experiment shows no VM or physical server failures. We are going to integrate a fault injector into our framework in the future. By setting the failure mode of cloud objects, our framework will be validated with more usage information and on more complex structures.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, Jun. 2009, pp. 599–616.

[2] G. DeCandia et al., "Dynamo: Amazon's Highly Available Key-value Store," ACM SIGOPS Oper. Syst. Rev., vol. 41, no. 6, Oct. 2007, p. 205–220.

[3] W. Zhao, P. M. Melliar-Smith, and L. E. Moser, "Fault Tolerance Middleware for Cloud Computing," Proc. of IEEE 3rd International Conference on Cloud Computing, Jul. 2010, pp. 67–74.

[4] M. Almorsy, J. Grundy, I. Müller, "An analysis of the cloud computing security problem," Proc. of APSEC 2010 Cloud Workshop, Sydney, Australia, Nov. 2010, pp. 109–114.

[5] Z. Zheng, T. C. Zhou, M. Lyu, and I. King, "Component Ranking for Fault-Tolerant Cloud Applications," IEEE Transaction on Service Computing, vol. 5, no. 4, Jul. 2011, pp. 540–550.

[6] Y. Dai, B. Yang, J. Dongarra, and G. Zhang, "Cloud Service Reliability : Modeling and Analysis," Proc. of the 15th IEEE Pacific Rim International Symposium on Dependable Computing, 2009.

[7] T. Thanakornworakij, R. F. Nassar, and C. Leangsuksun, "A Reliability Model for Cloud Computing for High Performance Computing Applications," Proc. of the 18th International Conference on Parallel Processing Workshops, Aug. 2012, pp. 474–483.

[8] Y. Tamura, M. Kawakami, and S. Yamada, "Reliability modeling and analysis for open source cloud computing," Proc. of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, vol. 227, no. 2, Apr. 2013, pp. 179–186.

[9] IEEE, "IEEE Standard Glossary of Software Engineering Terminology," IEEE Std. 610.12-1990, IEEE, 1990, pp. 1–84.

[10] V. Cortellessa and V. Grassi, "Reliability Modeling and Analysis of Service-Oriented Architectures," Test and Analysis of Web Services, pp. 339–362, 2007.

[11] M. R. Lyu, Handbook of Software Reliability Engineering, IEEE Computer Society Press and McGraw- Hill, pp. 574, 1996

[12] http://www.getcloudify.org/, [retrieved: Nov. 2014]

[13] http://www.openstack.org/, [retrieved: Nov. 2014]

[14] http://www.ganglia.info, [retrieved: Nov. 2014]

[15] http://httpd.apache.org/, [retrieved: Nov. 2014]

[16] http://tomcat.apache.org/, [retrieved: Nov. 2014]

[17] http://www.mongodb.org/, [retrieved: Nov. 2014]

 ISBN: 978-1-61208-388-9

Resource Self-management under an SLA within a Cloud Networking Environment

Mohamad Hamze, Nader Mbarek, Olivier Togni

Le2i Laboratory UMR 6306 CNRS
University of Burgundy
Dijon, France
e-mail: {Mohamad.Hamze, Nader.Mbarek, Olivier.Togni}@u-bourgogne.fr

***Abstract*—Today, cloud networking is one of the recent research areas in the cloud computing research communities. The main drawback of cloud networking consists in the lack of Quality of Service (QoS) guarantee and management in conformance with a corresponding Service Level Agreement (SLA). In this paper, we propose a framework for self-establishing an end-to-end SLA between a Cloud Service User (CSU) and several Cloud Service Providers (CSPs) in a cloud networking environment (inter-cloud Broker and Federation architecture). Then, we propose the self-management of cloud resources under the established SLA using specific autonomic cloud managers. We simulate our proposed framework to provide videoconferencing and intensive computing applications with self-management and QoS guarantee. We observe that the Broker architecture is the most economical, while ensuring QoS requirements.**

Keywords–Cloud Networking; Autonomic Computing; Self-management; Service Level Agreement; Quality of Service.

I. INTRODUCTION

Cloud computing is a promising technology for the realization of large, scalable and on-demand computing infrastructures. Many enterprises are adopting this technology to achieve high performance and scalability for their applications while maintaining low cost. However, a CSU that can be an end-user, an organization, a Software as a Service (SaaS) provider, or a Platform as a Service (PaaS) provider, requires for its services an end-to-end QoS guarantee with a high level reliability and a continued availability. In addition, cloud computing success requires that CSUs and CSPs can be confident that established SLAs are supporting their respective business activities to their best extent. However, an SLA may be violated when using a single CSP model due to unpredictable workload, resource failure and security attack.

Thus, geographical distributed data centers offer better end-to-end performance between CSU and CSP, while improving reliability when failure occurs. However, the inter-cloud should be designed as a multi-vendor environment with the ability to migrate services from one provider to another and to locate the best resources not only in terms of computing capacity and storage, but also connectivity, bandwidth and delay. Thus, the networking aspect of cloud computing is a critical factor.

In this context, cloud networking is defined as the ability to connect the user to his cloud services and interconnect services within an inter-cloud. It is built upon two main concepts, the integration of the networking resources onto existing data centers and the deployment of distributed computing and storage resources in the network [1]. It is difficult to guarantee the QoS of data transfer in cloud networking [2]. Thus, a convenient solution is to use a Bandwidth on Demand (BoD) service provided by a Network as as Service (NaaS) CSP. In such an environment, the development of an autonomic cloud control is necessary to simplify the complexity, maximize efficiency and minimize user interactions.

In this paper, we propose a framework for self-establishing an end-to-end SLA and self-managing CSU resources in a cloud networking environment thanks to the specification of an autonomic cloud networking architecture and the Autonomic cloud Managers description with their interactions and lifecycle. Then, we enable videoconferencing and intensive computing applications to take full advantage of our framework.

The remainder of this paper is organized as follows: Section II highlights the most relevant research works and trends in this area. In Section III, we present an overview of our proposed cloud networking architecture. In Section IV, we describe our autonomic cloud networking framework including autonomic cloud managers, SLA self-establishment and self-management lifecycle with cost calculation. Section V presents videoconferencing and intensive computing usage cases and the framework evaluation. Lastly, Section VI concludes the paper and points out future work.

II. RELATED WORK

The research project Scalable & Adaptive Internet Solutions (SAIL) [3] describes a cloud networking architecture and focuses on security, but it does not consider the QoS and SLA. The research project Foundation of Self-governing ICT Infrastructures (FoSII) [4] is proposing solutions for autonomic management of SLAs in the cloud. In addition, the research project Contrail [5] aims to vertically integrate an open-source distributed operating system for autonomous resource management in Infrastructure as a Service (IaaS) environments and PaaS systems. We consider our work to be very much in alignment with the objectives of these projects. But, our research work is innovative by considering the resource self-management under a self-established SLA in a cloud networking environment, and by focusing on minimizing service cost with QoS guarantee for IaaS and NaaS services.

From standardization perspective, IEEE Cloud Computing formed the Inter-Cloud Working Group (ICWG). It announced the launch of two new standards development projects: P2301 [6], a guide for Cloud Portability and Interoperability Profiles (CPIP) and P2302 [7], a Standard for Intercloud Interoperability and Federation (SIIF). Open Grid Forum (OGF) is active in the definition of the Open Cloud Computing Interface (OCCI) [8] for the interoperability between clouds. Global Inter-Cloud

 ISBN: 978-1-61208-388-9

Technology Forum (GICTF) [9] studies the standard Inter-Cloud interfaces to improve the reliability of the Clouds, and presents SLA metrics for Inter-Cloud environments. IBM presented in 2011 CloudNaaS [10], a cloud networking platform for enterprise applications. In our research work, we propose to develop a cloud networking framework and we aim to enable communications not only between CSPs Data Centers (DC), but also between CSU, CSPs (DC) and CSPs offering BoD. For that purpose, we use Web Services standard technologies.

Finally, there are many related research works on QoS [11][12], but QoS mentioned in these works is for SaaS, PaaS or IaaS. Smit et al. [13] present a methodology for an implementation of a service-oriented application that provides relevant metadata information describing offered cloud services via a uniform RESTful web service. In addition, several research works present SLA for cloud computing only: the project Mycloud [14] proposes Cloud Service Level Agreement (CSLA) and Patel et al. [15] propose to use Web Service Level agreement (WSLA) [16] in a cloud computing context. However, our research work considers, in addition, NaaS services and an SLA within different QoS attributes for autonomic cloud networking environment.

III. Cloud Networking Architecture Overview

A. Architecture Description

We consider two kinds of architectures, one for inter-cloud Broker and the other for inter-cloud Federation. We assume an environment with multiple CSPs interconnected together. Within these architectures, we ensure consistency between the QoS requirements requested by the CSUs, and service levels proposed by CSPs to allow multiple CSPs working together to meet the CSU requirements.

In the proposed cloud networking Broker architecture (Figure 1), the Cloud Broker is emerged as an intermediate entity between a CSU and CSPs to help the establishing of a service level that meets the CSU requirements. In addition, we have proposed two kinds of CSPs. The first one is the CSP (BoD) providing BoD network service (e.g., network operator) and playing the role of a NaaS CSP. The second one is the CSP (DC) providing IaaS and NaaS services. The IaaS service concerns Virtual Machine (VM) and storage resources and the NaaS service concerns network DC resources. The CSP (DC) can offer resources from one or several data centers. Moreover, CSPs can offer different service levels for example (Platinum, Gold, Silver, and Bronze), each one with different QoS guarantee and cost.

However, in the proposed cloud networking federation architecture, the federation provides an alliance among several CSPs (DC/BoD) that join together to help the establishing of service level that meet the CSU requirements. In addition, CSPs provide IaaS and/or NaaS services with different service levels. Furthermore, we consider that the CSU is connected to a Lead Cloud Service Provider CSP_L.

Within the Broker architecture, we consider three types of SLA constructed using the XML language for interoperability and portability between different entities:

1) inter-cloud Service Level Agreement (iSLA): it is a contract between a CSU and a Cloud Broker. It guarantees QoS for NaaS (BoD and/or DC) and/or IaaS (VMs and/or storage) services. QoS parameters could be quantitative or qualitative. In addition, it contains cost when violations occur.

2) BoD inter-cloud Service Level Agreement (B_iSLA): it is a contract between a cloud Broker and a CSP (BoD) interconnecting CSU sites, CSPs (DC) or connecting CSU sites to CSP (DC). It guarantees QoS for NaaS (BoD) services.

3) Datacenter inter-cloud Service Level Agreement (D_iSLA): it is a contract between a cloud Broker and a CSP (DC) for NaaS and IaaS services.

Moreover, in the Federation architecture, we consider the same three types of SLA. However, the iSLA is a contract between CSU and CSP_L, the D_iSLA is a contract between CSP_L (DC) and CSPs (DC), and the B_iSLA is a contract between CSPs (BoD) or between CSPs (DC) and CSPs (BoD) that enable CSU sites to reach them. In addition, we have two scenarios, the CSP_L can meet the CSU requirements without other CSPs resources usage (scenario 1) or using other CSPs resources in the alliance (scenario 2).

B. Problem Statement and Proposed Algorithms

1) Problem Statement: Constraint Optimization Problem: in general, the CSU requests IaaS services with or without NaaS services to run specific applications with QoS guarantee for its different sites using Brokerage or Federation services. Our goal is to minimize the cost subject to QoS Constraints. We must ensure a feasible and optimal CSPs selection. A feasible selection means that aggregated QoS values from selected CSPs satisfy the global CSU QoS requirements. Then, we consider as an optimal selection the feasible one minimizing the overall cost value (we propose two algorithms: Algs. 1 and 2). However, if we have the same minimum cost for different offers, our goal becomes to maximize a utility function for these offers to select the optimal one using different normalized weights w_i assigned by the CSU for each i-th QoS parameter based on its importance and the type of application. For that purpose, we specify two algorithms (Algs. 1 and 2) [17] that ensure best CSPs (DC) selection offering IaaS resources with best path selection between CSU sites and selected CSPs (DC).

2) Optimization and Path Selection Algorithm: the Cloud Broker in Broker architecture or the CSP_L (scenario 1) with other CSPs (DC) (scenario 2) in Federation architecture select the best CSPs (BoD) in terms of minimal cost and NaaS QoS guarantee using a proposed optimization and path selection algorithm (Alg. 1), subject to NaaS QoS parameters constraints when the CSU requests IaaS with NaaS QoS guarantee, or subject to minimal cost in case of IaaS only. Indeed, Alg. 1 calculates and sorts the cost of different service levels combinations offered by CSPs (BoD) for each route between each site and a specific CSP (DC) while guaranteeing the CSU QoS requirements. Finally, Alg. 1 selects the route corresponding to the minimal cost value.

3) Optimization and Resource Selection Algorithm: when the CSU requests IaaS with/without NaaS services, it interacts with the Cloud Broker or the CSP_L specifying its requirements. Then, the Cloud Broker or the CSP_L selects best IaaS CSP resources in terms of minimal cost and IaaS QoS guarantee using a proposed optimization and resource selection algorithm (Alg. 2) subject to IaaS QoS parameters constraints. At first, using Alg. 1, the Cloud Broker in Broker architecture gets best routes between CSU sites and each CSP (DC), whereas in Federation architecture, only the CSP_L (scenario 1) or the latter with other CSPs (DC) (scenario 2) get best routes between CSU sites and themselves. Furthermore, the

 ISBN: 978-1-61208-388-9

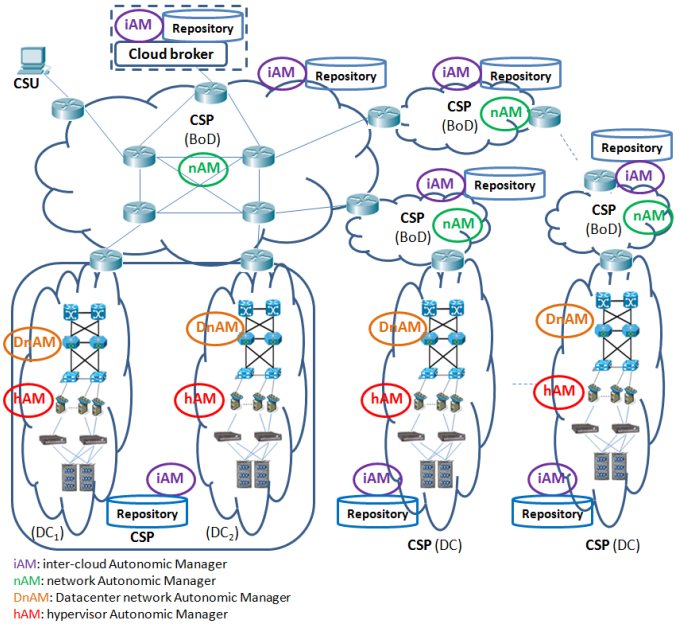


Figure 1. Autonomic Cloud Networking Architecture.

Cloud Broker or the CSP_L selects best resources that meet the CSU requirements while taking into account best routes (Alg. 1). Note that in the federation architecture, the CSP_L selects at first its available resources then it selects remaining resources in other CSPs (DC) if needed.

IV. Autonomic Cloud Networking Framework

Due to the on-demand self-service characteristic of cloud computing, it is required that a cloud infrastructure supports SLA self-establishment and resource self-management. In this section, we propose to self-manage resources and establish iSLA, B_iSLA and D_iSLA in our cloud networking architectures autonomously using Autonomic cloud Managers (AMs).

A. Autonomic Cloud Networking Architecture

In general, a domain refers to resource collections managed by a single entity, e.g., cloud Data Center (DC) or communication network. If we manage this domain in an autonomic manner, we call it Autonomic cloud Domain (AD). In this context, we present the proposed cloud networking architecture with several ADs (Cloud Broker or CSPs (DC/BoD)). Figure 1 represents this architecture including a cloud Broker entity for the Broker scenario and without it for the Federation scenario. Each AD is under the authority of an inter-cloud Autonomic Manager (iAM). iAM communicates with other iAMs to achieve an agreement on a service level. In addition, it controls one or more low level AMs to configure resources in conformance with the agreed service level. These AMs are playing different roles within our autonomic architecture:

1) network Autonomic Manager (nAM): it is responsible for creating and managing CSP (BoD) virtual networks and monitoring workload and performance in conformance with the agreed B_iSLA.

2) Datacenter network Autonomic Manager (DnAM): it is responsible for creating and managing CSU virtual networks within the cloud DC and monitoring workload and performance in conformance with the NaaS part of D_iSLA.

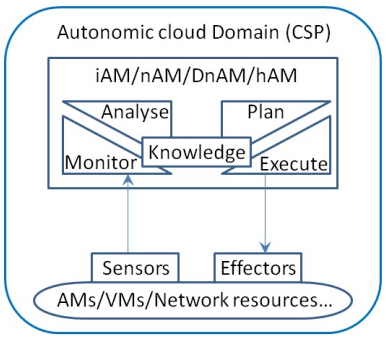


Figure 2. Autonomic cloud Manager Functional Details.

3) hypervisor Autonomic Manager (hAM): it is responsible for creating and managing VMs and storage capacities in conformance with the IaaS part of D_iSLA. Therefore, the CSP (DC) can consequently decide the allocation or deallocation of resources to maintain an acceptable performance.

We provide these autonomic cloud managers with the capability to achieve an agreement between ADs. This agreement covers QoS aspects for different cloud service models such as IaaS and NaaS with different service levels. In addition, the iAM entities use a repository to store resource management information and to facilitate their interactions with other AMs.

B. Autonomic cloud Manager Description

An Autonomic cloud Manager (Figure 2) has to know its environment and how to keep it in optimal conditions without the need of any external operation. Therefore, our proposed AM (iAM, nAM, DnAM, or hAM) can manage a single resource or set of resources (AMs, VMs, storage and network resources, etc.) thanks to sensors and effectors interfaces.

After integrating various policies in its knowledge base (thresholds, algorithms for best cloud resource selection, etc.), the AM begins with the monitoring phase (QoS parameter values, etc.) to ensure data collection, aggregation, filtering and reporting from managed resources thanks to sensor interfaces. Then, the collected data is passed to the analysis phase to correlate these data in accordance with the knowledge base policies (QoS parameter violation, failure, congestion, etc.). Then a request for a change could be sent to the planning phase to indicate actions needed to achieve specific objectives in accordance with specified policies (cloud resource allocation or releasing, etc.). Finally, these actions are sent to the execution phase that allows changes to be made in the managed cloud resource thanks to effectors interface (resource configuration, VM migration, etc.). In addition, the changes are checked to update the knowledge base by the monitoring phase. These phases constitute the closed control loop (MAPE-K) of cloud resource self-management implemented by our AMs.

C. Autonomic Cloud Managers Interactions

To provide our cloud networking architecture with autonomy while offering an end-to-end QoS guarantee, two kinds of interactions could take place between autonomic cloud managers (Figure 3 for Broker scenario and Figure 4 for Federation scenario). Thanks to the first one, an iAM initiates a peer to peer communication process with the corresponding

 ISBN: 978-1-61208-388-9

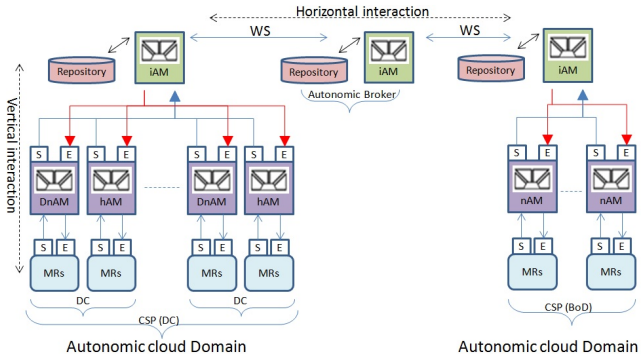


Figure 3. AM Interaction Framework for Broker scenario.

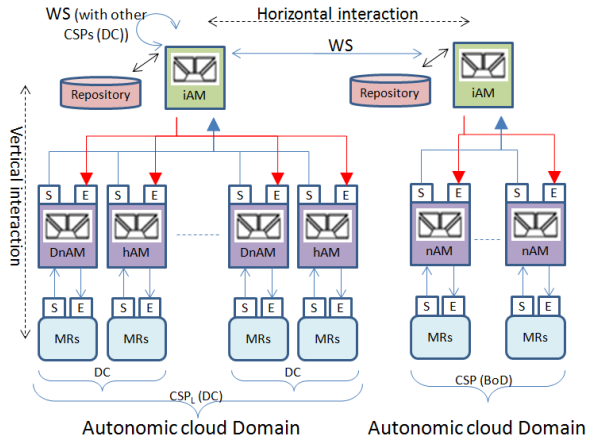


Figure 4. AM Interaction Framework for Federation scenario.

iAMs in a horizontal interaction using Web Services (WS) technologies to achieve an agreement on a service level. In addition, each iAM is responsible of the service level guarantee within the corresponding AD. This guarantee will be possible thanks to a second kind of interaction. Indeed, the iAM controls one or more low level Autonomic cloud Managers (nAM, DnAM or hAM) thanks to the manageability interfaces (effectors and sensors) using WS technologies to achieve this service level guarantee. Therefore, iAM provides these low level AMs with the corresponding service level (B_iSLA or D_iSLA) in a vertical interaction, so that they use a similar interaction to allocate, release, or modify the configuration of their Managed Resources (MRs: router, VMs, storage, etc.) according to the received service level.

D. SLA Self-establishment Lifecycle

The following Finite State Machines (FSMs) presents the lifecycle of the proposed AMs for SLA self-establishment:

1) Broker Architecture: The FSM concerning Broker iAM (Figure 5) includes three states. In the first state S0, the Broker iAM receives periodically available services with different service levels or any changes from CSPs (DC/BoD) iAMs in the alliance. Then, it updates its repository. After receiving CSU service requirements to construct an iSLA, the Broker iAM goes to the second state S1.

In state S1, the Broker iAM consults its repository and compares the CSU requirements with different services and

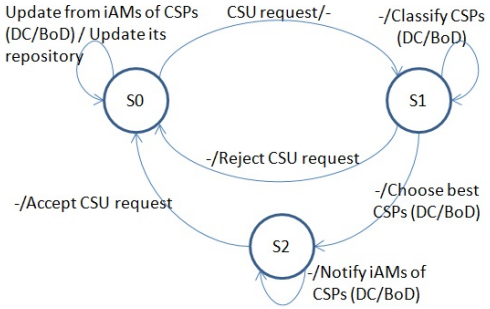


Figure 5. FSM for Broker iAM Lifecycle.

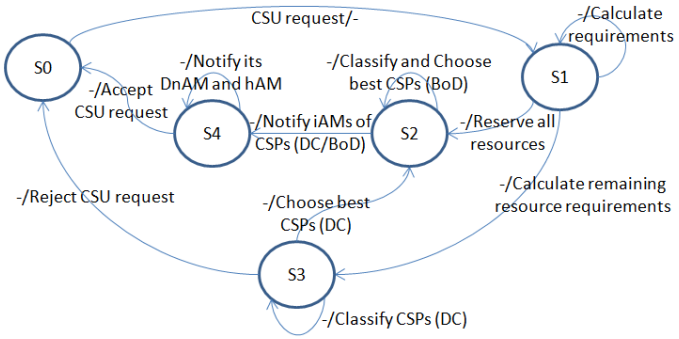


Figure 6. FSM for CSP_L iAM Lifecycle.

their corresponding service levels offered by CSPs iAMs to select the appropriate CSPs that meet the CSU QoS requirements according to the proposed optimization and selection algorithms (see Sec. III-B). If the Broker iAM does not find any CSPs that meet the CSU requirement, it rejects the CSU request and goes to the initial state S0. Else, the Broker iAM chooses the suitable CSPs, and goes to state S2.

In state S2, the Broker iAM notifies each iAM of selected CSPs and establishes a D_iSLA and a B_iSLA with respectively each CSP (DC) iAM and CSP (BoD) iAM. Then, CSPs (DC) iAMs and CSPs (BoD) iAMs provide their AMs with the corresponding service level to allocate resources and deliver IaaS and/or NaaS services with QoS guarantee according to the received service level (B_iSLA or D_iSLA). Finally, the Broker iAM accepts the CSU request, establishes the iSLA, and goes to the initial state S0.

2) Federation Architecture: The FSM that we specify for the CSP_L iAM (Figure 6) includes five states. In state S0, when the CSP_L iAM receives a CSU request, state S1 is reached to calculate resource requirements. In state S1, if the CSP_L can meet all CSU requirements, the corresponding CSP_L iAM goes to state S2 to reserve resources. Else, it calculates remaining resource requirements and goes to state S3. In this state, it contacts other CSPs (DC) iAMs in order to meet remaining CSU resource requirements. On the other hand, each iAM of these CSPs (DC) selects best CSPs (BoD) according to the proposed optimization and path selection algorithm. Also, it describes IaaS and NaaS services for available resources with different service levels and sends all information to the CSP_L iAM. Then, if the CSP_L iAM does not find any CSPs that meet requirements, it rejects the CSU request and goes to the initial state S0. Else, it selects best

 ISBN: 978-1-61208-388-9

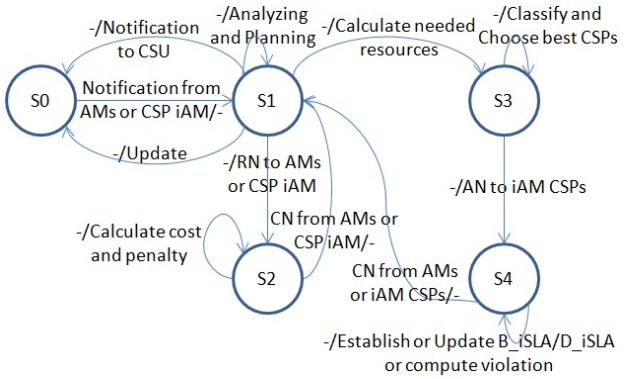


Figure 7. FSM of Broker or CSP_L iAM Control Loop Lifecycle.

CSPs that meet CSU QoS requirements for IaaS with/without NaaS services according to the proposed optimization and resource selection algorithm and goes to state S2.

In state S2, the CSP_L iAM classifies and selects best CSPs (BoD) that enable CSU sites to reach it according to the proposed optimization and path selection algorithm. Then, the CSP_L iAM only (scenario 1) or the latter with iAMs of selected CSPs (DC) (scenario 2) notify iAMs of selected CSPs (BoD) in order to allocate BoD network resources and establish B_iSLA with them. In addition, each iAM of CSPs (BoD) provides its nAM with the corresponding service level to deliver NaaS services with QoS guarantee. Moreover, in scenario 2, the CSP_L iAM sends a request to establish a D_iSLA with iAMs of selected CSPs (DC). Then, each concerned CSPs (DC) iAMs provide their DnAM and hAM with the corresponding service level to deliver IaaS and/or NaaS services with QoS guarantee.

After this, the CSP_L iAM goes to state S4 while notifying its hAM and DnAM to reserve and configure VMs, storage and network resources in order to deliver IaaS and NaaS services with QoS guarantee. Finally, the CSP_L iAM accepts the CSU request, establishes the iSLA, and goes to initial state S0.

E. Self-management Lifecycle

After the SLA self-establishment, the objective is to guarantee cloud service performances by periodically monitoring the CSU agreed SLA at CSPs using AMs sensors and policies defined by each AD to avoid SLA violation. In addition, the strategy of detecting SLA violations is based on the use of predefined violation thresholds specified by each AD. On the one hand, a violation threshold is a value indicating the least acceptable performance level for an application in conformance with the agreed SLA and real-time monitoring information. Therefore, exceeding the violation threshold value for a particular QoS parameter indicates the occurrence of SLA violation and the system logs the necessary information for calculating the appropriate penalties. On the other hand, a risk threshold is a value indicating the risk of a violation for a performance level. Therefore, with this information the system can react quickly to avoid the violation threat and save the CSP from costly SLA penalties. Moreover, each AD specifies a threshold that indicates if the CSU have exceeded the maximal traffic throughput defined in the iSLA.

When an unexpected surge of access to cloud services occurs, there is a risk of degradation in QoS parameters at one or more CSPs (DC/BoD). Therefore, when a risk concerning a QoS parameter violation is detected at a CSP, available resources in this CSP or in other CSPs must be autonomously discovered and reserved with self-optimization and self-reconfiguration functions to solve this problem thanks to the control loop and AMs interactions using our proposed optimization and selection algorithms. Thus, the cloud self-management process is described thanks to an FSM including five states as presented in Figure 7.

1) State S0: in this state, the Broker iAM in Broker architecture is waiting to receive a notification from an iAM of a CSP to go to the second state S1. However, in Federation architecture, the CSP_L iAM is waiting to receive a notification from an iAM of a CSP or from AMs under its control (nAM, DnAM or hAM) to go to the second state S1. This notification collected by iAMs sensors can be a service termination, a risk for QoS parameter violation, an exceeding of throughput threshold, or an update at the AD of a CSP_L or a CSP.

2) State S1: in this state, the Broker or CSP_L iAM analyzes the notification and plans to react with the appropriate actions. Thus, if the notification is a service termination, the Broker or the CSP_L iAM sends a Release Notification (RN) to the CSP_L AMs or to a CSP iAM according to the established B_iSLA or D_iSLA to release resources and goes to state S2. However, if the notification is a risk of QoS parameter violation based on the QoS parameter threshold, the Broker or the CSP_L iAM can resolve the problem by allocating new resources free of charge to avoid violation. Therefore, it calculates needed resources and goes to state S3. Note that, we consider that a CSP cannot react only to a risk of QoS parameter violation to resolve the problem. Otherwise, it resolves this problem by allocating new resources in its AD in conformance with the established D_iSLA or B_iSLA. Then, it sends an update notification to the Broker or the CSP_L iAM that updates its repository and goes to state S0.

3) State S2: in this state, the Broker or the CSP_L iAM calculates the cost of terminated resources and penalties based on violations and the established B_iSLA or D_iSLA. Then, the CSP_L AMs or a CSP iAM release these resources using vertical AMs interaction and the corresponding CSP pays costs to the Cloud Broker or to the CSP_L and sends a Confirmation Notification (CN) to the Broker or the CSP_L iAM. Next, the Broker or the CSP_L iAM goes to the state S1.

4) State S3: in this state, the Broker or CSP_L iAM selects best CSPs based on the proposed optimization and selection algorithms in conformance with the established iSLA. Then, it sends an Allocation Notification (AN) to each iAM of selected CSPs to allocate resources and goes to the state S4.

5) State S4: in this state, the Broker or CSP_L iAM establishes or updates the B_iSLA or the D_iSLA. In addition, when the Broker or CSP_L iAM cannot avoid the violation, it considers this violation to calculate the penalty when the service is terminated. Moreover, the CSP iAM, CSP_L AMs or each iAM of selected CSPs send a CN to the Broker or CSP_L iAM that enabling their transition to state S1.

The Broker or CSP_L iAM in state S1 analyzes the new state of resources. If a problem is detected, the Broker or the CSP_L iAM tries to plan and resolve this problem. Else, it updates its repository (Knowledge base) with changes, notifies the CSU for resource allocation or releasing, and goes to the

 ISBN: 978-1-61208-388-9

initial state S0. Note that, if some CSU requests exceed the corresponding throughput threshold, the CSP iAM can drop or delay these requests or allocate new paid resources to the CSU based on the established iSLA. Then, it sends an exceeding of throughput notification to the Broker or the CSP_L iAM that allocates new paid resources to the CSU if needed.

F. Cost Calculation

This section introduces a general methodology to calculate costs (1) similar to the way Amazon is charging its clients.

$$Cost_{total} = Cost_{VM} + Cost_{BW} \quad (1)$$

where, $Cost_{total}$ is the total cost of different CSU resource consumption, $Cost_{VM}$ is the total cost of VMs resources (2) and $Cost_{BW}$ is the total cost of bandwidth resources (3).

$$Cost_{VM} = \sum_{j} \sum_{i} (VMc_{ij} \times t_i) \quad (2)$$

$$Cost_{BW} = \sum_{k} (BWc_k \times BW_k) \quad (3)$$

where, VMc_{ij} ($ per hour) is the cost unit of a selected VM type vt_i in a selected CSP_j (DC), and t_i is the number of vt_i consumption hours. BWc_k ($ per GB) is the cost unit of the traffic traversing a selected CSP_k (DC and/or BoD) with a QoS level and BW_k (GB) is the CSU traffic traversing the selected CSP_k.

V. Usage Cases and Evaluation

To take full advantage of our proposed autonomic cloud networking framework, we present in this section two usage cases. The first one is for a large-scale cloud videoconferencing application which is one of the most demanding multimedia applications in terms of bandwidth, end-to-end delay and jitter QoS parameters. The second usage case is for intensive computing application which is based on requests for the execution of computationally intensive tasks. Therefore, we use our autonomic framework to self-establish SLAs and self-manage these applications while minimizing the total application cost without violating end-to-end QoS parameters.

We test corresponding usage cases as a proof of concept and evaluate performances by conducting a set of simulations using the CloudSim toolkit [18]. We extend CloudSim to support three new entities. The first one is a CSU entity and the second is a Cloud Broker entity, instead of DatacenterBroker entity. In addition, we propose a CSP (BoD) entity that interconnects the Broker, CSPs (DC) and CSUs using BRITE topology [18] for modeling link bandwidth and latencies. In addition, we use the Sensor class for monitoring resources and specifying thresholds and the SimEven class for specifying notifications. Therefore, the control loop algorithms at each entity constitute the iAM, nAM, DnAM and hAM.

The simulated model is composed of one Broker, four CSPs (BoD) and four CSPs (DC) containing each one 10 hosts. A host has quad-core processors (4×1,2GHz) and 16GB of RAM. All entities are initiated at the beginning of the simulation. For a Gold service level for example, a CSP (DC) can have the following characteristics in our simulation model: 750 MHz VM CPU capacity, 99.999% IaaS availability, NaaS QoS parameters: {Latency 7 ms, Jitter 1 ms, Packet Loss Ratio 2.5×10^{-3}, Bandwidth 10Mb/s, NaaS availability 99.999%}, Bandwidth cost 0.05 ($ per GB) and CPU cost 0.3 ($ per hour). In addition, a CSP (BoD) can have the following characteristics: NaaS QoS parameters: {Latency 12 ms, Jitter 2 ms, Packet Loss Ratio 2.5×10^{-3}, Bandwidth 10Mb/s, NaaS availability 99.999%} and Bandwidth cost 0.3 ($ per GB). Other CSPs (DC/BoD) have different values for each service level (Platinum, Gold, Silver, Bronze).

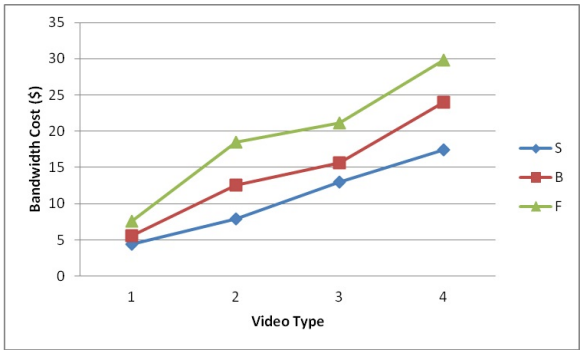


Figure 8. Global bandwidth cost comparison.

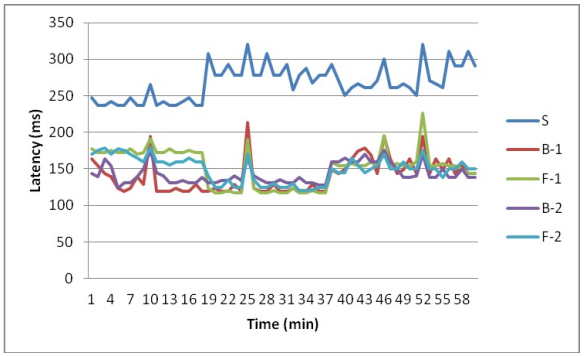


Figure 9. Global end-to-end latency comparison.

In each usage case, we evaluate three simulation scenarios. The first one corresponds to a static selection of resources without SLA self-establishment and QoS guarantee. The second scenario is a broker and a federation based architecture with SLA self-establishment and QoS guarantee but without resource self-management, i.e., in case of violation the system will not react. The third scenario is the same as the second one but with resource self-management. In each scenario, we have multiple CSU sites connected to different CSPs (BoD) that use IaaS with NaaS services in broker and federation scenarios. In addition, the violation is simulated as a traffic sent by another party that affects QoS parameters of CSU resources.

A. Usage Case 1: Cloud videoconferencing scenario

We simulate 4 video types with one hour length, a size of 1, 2, 3 and 4 GB respectively and a bandwidth of 2.2Mb/s, 4.5Mb/s, 6.8Mb/s and 9.1Mb/s, respectively. The CSU specifies the latency less than 180 ms in Broker and Federation architectures. We calculate the global bandwidth cost of each video type while comparing the first and the third scenarios (Figure 8). In addition, we calculate the global network latency of the first video type, with a sampling interval period of one minute, to compare the three scenarios (Figure 9).

As shown in Figure 8, the bandwidth cost increase when the video bandwidth increases. Moreover, in a static selection (S),

 ISBN: 978-1-61208-388-9

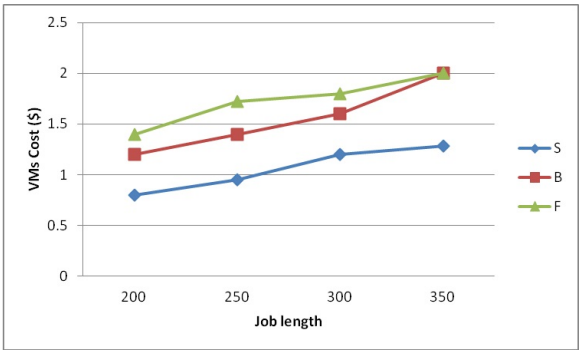


Figure 10. Global VMs cost comparison.

Figure 11. Global response time comparison.

the bandwidth cost is less than the Broker (B) and Federation (F) costs due to the NaaS QoS guarantee. Furthermore, the bandwidth cost in the Broker scenario is less than the Federation scenario due to the selection of resources in the CSP_L firstly. Therefore, CSP_L resources are not usually the best and so the Broker architecture is the most economical while ensuring QoS requirements. In addition, as shown in Figure 9, the results reveal a good streaming latency achieved by our Broker (B-1/2) and Federation (F-1/2) proposal, as compared to a static (S) selection. However, in case of violation, the Broker or the CSP_L (B/F-2) can react and avoid the violation as compared to results without self management (B/F-1).

B. Usage Case 2: Cloud Intensive Computing

We simulate different job lengths (200, 250, 300 and 350 instructions) that are executed by different VMs. For each job length, the CSU sends 10 jobs to four VMs. The CSU specifies a response time less than 300 ms. We calculate the global VMs cost of each job length during an hour while comparing the first and the third scenarios (Figure 10) and the global average response time to compare the three scenarios (Figure 11).

As shown in Figure 10, the VMs cost increase when the job length increases. Moreover, the VMs cost in static selection is less than the VMs cost in Broker and Federation scenarios due to the IaaS QoS guarantee. Furthermore, the VMs cost in the Broker scenario are less important than the Federation scenario according to the same reasons mentioned in usage case 1. Thus, the Broker architecture is the most economical. In addition, as shown in Figure 11, the response time is well controlled (B/F-1/2) as compared to a static selection (S) thanks to IaaS QoS guarantee. However, in case of violation, the Broker or the CSP_L (B/F-2) can react and avoid the violation as compared to results (ex. Job 5/16/37) without self management (B/F-1).

VI. Conclusion and Future Work

In this paper, we presented a framework for self-establishing an SLA and self-managing CSU resources within cloud networking environment. At first, we have proposed an autonomic cloud networking architecture for Broker and Federation scenarios. In addition, we have presented the description, interactions and the lifecycle of our autonomic cloud managers. Finally, we have evaluated our proposed framework for cloud videoconferencing and intensive computing applications and we have obtained good performance results. We observed that the Broker architecture is the most economical.

As a future work, we aim to define the penalty calculation in the SLA. In addition, security parameters will be included in our proposed iSLA to provide our autonomic cloud networking framework with self-protection and study their impact on QoS parameters.

References

[1] P. Murray, A. Sefidcon, R. Steinert, V. Fusenig, and J. Carapinha, "Cloud Networking: An Infrastructure Service Architecture for the Wide Area," in Proceedings of the 21st Annual Conference on Future Network & Mobile Summit, 2012, pp. 1–8.

[2] ITU-T, "Requirements and framework architecture of cloud infrastructure," in Focus Group on Cloud Computing, TR part 3, 2012, pp. 1–59.

[3] "SAIL Project," URL: http://www.sail-project.eu/ [retrieved: 01, 2015].

[4] "FoSII," URL: http://www.infosys.tuwien.ac.at/linksites/FOSII/index.html [retrieved: 01, 2015].

[5] "Contrail-project," URL: http://contrail-project.eu/ [retrieved: 01, 2015].

[6] "CPIP," URL: http://standards.ieee.org/develop/project/2301.html [retrieved: 01, 2015].

[7] "SIIF," URL: http://standards.ieee.org/develop/project/2302.html [retrieved: 01, 2015].

[8] "OCCI," URL: http://occi-wg.org/ [retrieved: 01, 2015].

[9] "GICTF," URL: http://www.gictf.jp/index_e.html [retrieved: 01, 2015].

[10] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "CloudNaaS: a cloud networking platform for enterprise applications," in Proceedings of the 2nd ACM Symposium on Cloud Computing, 2011, pp. 1–13.

[11] Z. Zibin, Z. Xinmiao, W. an Yilei, and M. Lyu, "QoS Ranking Prediction for Cloud Services," IEEE Transaction on Parallel and Distributed Systems, vol. 24, 2013, pp. 34–47.

[12] R. Karim, D. Chen, and A. Miri, "An End-to-End QoS Mapping Approach for Cloud Service Selection," in IEEE Ninth World Congress on Services, 2013, pp. 341–348.

[13] M. Smit, B. Pawluk, P. ad Simmons, and M. Litoiu, "A Web Service for Cloud Metadata," in IEEE Eighth World Congress on Services, 2012, pp. 361–368.

[14] Y. Kouki and T. Ledoux, "CSLA: a Language for improving Cloud SLA Management," in Proceedings of the International Conference on Cloud Computing and Services Science, 2012, pp. 586–591.

[15] P. Patel, A. Ranabahu, and A. Sheth, "Service Level Agreement in Cloud Computing," in Proceedings of the OOPSLA Cloud Computing workshop, 2009, pp. 1–10.

[16] H. Ludwig, A. Keller, A. Dan, R. King, and R. Franck, "Web service level agreement (WSLA) language specification," IBM Corporation, 2003, pp. 815–824, version 1.0.

[17] M. Hamze, N. Mbarek, and O. Togni, "Autonomic Brokerage Service for an End-to-End Cloud Networking Service Level Agreement," in IEEE 3rd Symposium on Network Cloud Computing and Applications, NCCA, 2014, pp. 54–61.

[18] R. Calheiros, R. Ranjan, A. Beloglazov, C. Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environment and evaluation of resource provisioning algorithms," Software: Practice and Experience, vol. 41, 2011, pp. 23–50.

 ISBN: 978-1-61208-388-9

Autonomic Management for Energy Efficient Data Centers

Forough Norouzi
Computer Science department, Western University
London, Canada
e-mail: fnorouz@uwo.ca

Michael Bauer
Computer Science department, Western University
London, Canada
e-mail: bauer@uwo.ca

***Abstract*—The complexity of today's data centers has led researchers to investigate ways in using autonomic methods for data center management. In this work, we consider using autonomic management techniques that can help reduce data center energy consumption. In particular, we consider policy-based, multi-level autonomic management for energy aware data centers. We advocate for a hierarchical model of managers with loosely coupled communication between them. We describe our manager topology, communications and manager operations. We implement our approach for high performance computing centers that may have one or more large high performance computing systems. A data center simulator has been implemented that calculates data center energy consumption. We evaluate different management policies and our approach using this simulator. Preliminary experiments show promising results in terms of minimizing energy consumption and overhead on service level expectations in high performance computing systems.**

Keywords- autonomic computing; energy aware data center; self-management system; policy-based management.

I. INTRODUCTION

Today's data centers are large, complex and challenging to manage. One of the central challenges in data center management and operations is energy management. Data centers at the core of Internet-scale applications consume about 1.3% of the worldwide electricity supply, and this level is predicted to increase to 8% by 2020 [1]. Google alone, for example, consumed 2.26M MWh in 2010 [4]. Carbon emissions from data centers alone in November 2008 were 0.6% of the global total and predicted to be 2.6% by 2020 which is more than the total carbon emission of Germany [3]. Given these statistics, reducing the energy consumption of data centers and making them work in an energy-aware manner is a major topic of data center management research. Broadly, research into energy efficiency in data centers can be categorized into a number of areas. Server level energy management approaches take advantage of lower power states built into components e.g. CPU(Central Processor Unit) and memory. At the level of clusters, management models aim to use optimization and control theoretic approaches to optimize the number of required compute node for each running application. Virtualization looks at reducing the number of active physical servers by multiplexing them as virtual machines (VM) where having fewer physical servers means that other servers can be turned off or maintained in a low power state. Thermal aware scheduling considers energy consumption criteria for job scheduling and resource allocation. However, there are few approaches looking into overall holistic strategies and automated methods to support administrators. One strategy is to consider approaches based on autonomic management, particularly policy-based autonomic management, where part of the role of the administrator would be codifying management policy for data center operations. Autonomic Computing (AC) aims to embrace the notion of self-management in distributed and complex systems where administrator intervention in system management is reduced or minimized. Instead, administrators define the overall policy and strategy for system management according to system organizational objectives. Self-management based on use of policies is referred to as *policy-based management*; it is a promising approach for developing autonomic management in complex distributed systems.

We advocate for multiple autonomic managers rather than having a single centralized autonomic manager that could be a single point of failure and potential performance bottleneck. To the best of our knowledge, policy-based autonomic management utilizing multiple managers for energy aware data centers is only marginally addressed in previous research. The proposed management system focuses on multilateral interaction in a multi-agent autonomic computing environment where autonomic managers interact with each other in a hierarchical structure. Intuitively, a hierarchical arrangement of managers would seem to provide good scalability while keeping communication overhead low and some previous research has suggested the utility of hierarchical management [4][14]. A hierarchical approach also matches well the hierarchy of computational elements in the data center.

This paper organized as follows. Section 2 provides an overview of related work. Aspects of data center management and our proposed management system architecture will follow. The data center simulator and a number of implementation scenarios for a simple data center are in Section 5. We conclude with a discussion on management overhead and future plans.

II. RELATED WORK

Autonomic Computing (AC) refers to the idea of a computing system or application being self-managing, that is, a system that can manage itself in such a way that it is adaptable to any changes in the system environment [6]. In the autonomic computing paradigm, a management module

 ISBN: 978-1-61208-388-9

which controls the behavior of a managed element (ME) is called an autonomic manager (AM). The managed element provides some sensors and actuators to the manager. The manager monitors available metrics through these sensors and analyses the monitored information. It can then plan for a series of actions that need to be taken, if any, and execute those actions through the provided actuators. This process is a feedback loop called the Monitor-Analyze-Plan-Execute (MAPE) loop [10]. In AC, different AMs control different resources in a distributed manner. This management could be done individually, i.e. each AM is responsible for its own MEs. More generally, in computing systems it is necessary that AMs interoperate. There may be heterogeneous types of AMs that may have different objectives. Research by Mukherjee [13] illustrates coordination between two independent AMs where the first AM deals with service level agreement (SLA) management and resource allocation, while the second AM deals with minimizing power consumption by turning off unused servers. Their work shows that the interaction between the managers is important in achieving the goals.

Khargharia et al. [5] introduced a three-level hierarchy for optimizing energy consumption and SLA violations. The hierarchy starts from the device level inside a server, proceeds to the server level and then encompasses the cluster level. Decisions are based on the power status of each managed element at each level. Their idea illustrates the value of a hierarchical approach, but needs some modification to be applicable for large scale data centers which may have many different types of applications and services. Anthony et al. [26] identify collaboration as a key aspect and suggest that AMs should be designed for collaboration and that the lack of collaboration between managers is a problem. Then, the authors attempt to tackle AM interoperability issues and define an interoperability service. The interoperability service keeps a database of registered AMs along with corresponding resources they manage and scope of their management operation. The interoperability service will detect potential conflicts and send messages to related AMs to, for example, suspend or stop their activities. Kusic et. al. [16] described an autonomic cluster management framework. They defined three different types of agents: general agents (implemented per node), optimization agents, and configuration agents (implemented per implementation of the management framework). The proposed management infrastructure is a hybrid of centralized and decentralized and communication between agents is done via message passing. Complexity of task distribution between agents makes it un-scalable for large scale environment. Kennedy [9] argues that the mechanism that defines interoperability between autonomic elements must be reusable and generic enough to prevent complexities. A standard means must be defined to exchange context between autonomic elements. This meta level needs to be context-aware. At this step, they have identified the main challenges for automated recovery in autonomic system. Thomas, et. al. [22] presented a management framework for the automated maintenance cycle in the computing cluster (part of the Data Grid project [23]). A number of management modules, e.g. job management, monitoring, fault recovery, and configuration management, have been defined where each produces information as an output which is used as input for others. Their system gets configuration states from an administrator. Each machine has a goal state which is stored in a configuration database and also has an actual state which comes from the monitoring agent. These states are compared within the fault detection and recovery system for any mismatch, which then applies any necessary actions to fix them. The fault detection system has its set of rules (policies) for each node where these rules are checked. A decentralized architecture, Unity, was introduced in [24]. Unity introduces a two level management model that tries to allocate optimized resources (servers) to different types of application environments running both batch type and interactive workload across the whole data center.

Policy Based Management (PBM) is a management paradigm that separates governing rules from the main functionality of the managed system. Bahati et al. [18] described an architecture for autonomic management and demonstrated how policies are defined and mapped to their corresponding elements. The authors of [13] propose a Model-driven Coordinated Management architecture to make dynamic management decisions based on energy benefits of different policies to handle events. They used a workload model, power model, and thermal model to predict the impact of different management policies. A central management unit monitors events, chooses the best policy and makes decisions.

III. A MODEL FOR DATA CENTER MANAGEMENT SYSTEM

One can think of the AMs and their relationships as a kind of management overlay network on top of the elements of the data center. The actual position of management modules might be on single physical server, or even distributed over a number of servers. Number of essential questions need to be addressed before implementing the management system. For instance: what are managed objects in the data center? What metrics of an object should or could be monitored? And what are possible actions that the management system could take to control that specific object? To develop a management system, which contains a dynamic number of managers for a data center, several issues need to be addressed:

Topology of the AMs: AMs are more likely to have their own overlay network, with a specific protocol to communicate and exchange information e.g. SOAP (Simple Object Access protocol). The topology has implications for the coordination and communication among AMs and the decomposition of management tasks among AMs.

- Hierarchical management means that some AMs can monitor and influence or control the behavior of other AMs. In this case, lower AMs are considered as managed elements for the higher AM. AMs at different levels usually work at different time scales. In this topology the upper layer AM regulates and orchestrates the system by monitoring parameters of all of its lower level AMs (see Figure 1). The upper layer AM is privileged over lower

 ISBN: 978-1-61208-388-9

layer AMs, and has the authority to control or manipulate some parameters of the lower level AMs.

- A peer to peer topology entails AMs that can directly communicate with one another, exchange information and make decisions. In this paradigm, all AMs are often equally privileged.
- Indirect coordination between AMs involves an AM making changes in its MEs which are then sensed by other AMs causing them to perform actions. There is no direct communication between the AMs. Since the MEs (e.g. application, services, and virtual machine) may change over time (e.g. is finished or started), there should be a way such that the topologies of corresponding AMs can change on the go.

Collaboration Strategy: Depending on the topology, the next question is how AMs influence other AMs in the management system? How much information do they need to share? What kind of information? For example, one AM may be privileged over some set of other AMs because its management scope is wider than the others or it has more information about its surrounding environment. Alternatively, all AMs could be acting the same, e.g. as in a peer-to-peer topology. Finally, what is the nature of AMs interaction and coordination?

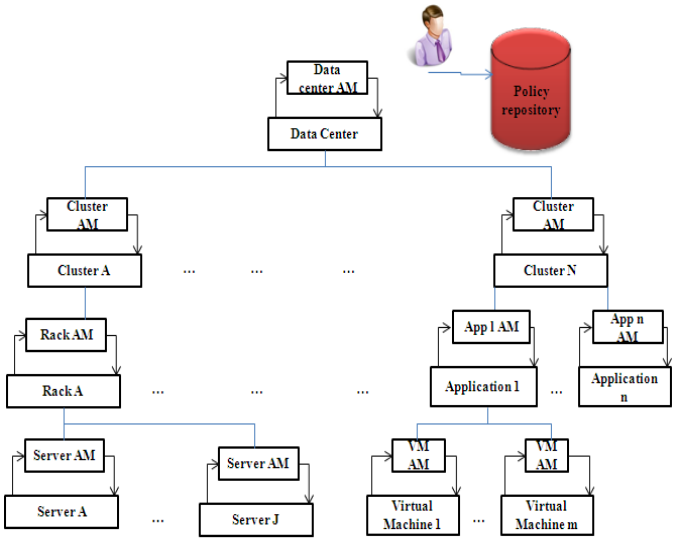


Figure 1. Hierarchical Policy-based Management. System.

Manager Life Cycle: An autonomic manager has its own life cycle which obviously corresponds to the life cycle of its associated MEs. For example, for a cloud user renting compute nodes and running an application for a period of a time, the corresponding AM is born and dies along with the application life cycle. One of the issues in multilevel management systems is that each level of the management model has to have the ability to create AMs based on the respective ME life cycle, then introduce it to the management system, and then destroy it at the end.

The overarching management approach assumes that the management system will associate an AM with the new arrival ME; this could be done automatically if the administrator specified a particular AM for that class of ME or could be just done manually by the administrator. We expect that in many cases, the MEs can be grouped into broad classes and, correspondingly, that AMs will be as well. In our approach, we assume classes of AMs that have similar requirements and characteristics will be defined and associated with classes of MEs. Table I. illustrates samples of MEs and corresponding classes of AMs. The management system refers to this table upon the initialization

TABLE I. CLASSES OF AMS AND MES

ME class	Generic associated AM
VM	VM_AM.class
Cluster	clusterAM.class
Rack	rackAM.class
Application -interactive -Enterprise	 appIAM.class appEAM.class

or creation of a new ME to check which class of AM should be initiated for that particular ME. Part of the AM initialization is to identify its parent and to get its policies.

The proposed management system is policy-based which means that each AM has its own set of ECA (Event Condition Action) policies referred to as a policy profile. This set can be altered according to the system situation or even a direct change from the data center administrator. In our model, the parent AM can also make decisions regarding the policy profile of its children as part of its planning task. Policy repository holds the policy profiles associated with each of the managed element classes. Generally, though, we would expect that there would be much overlap, e.g. a policy for managing an application during a work day would be very similar to the policies for managing it at night or on a weekend.

A. *Management System Configuration*

An administrator first needs to decide about the number of management levels in the management system and then the position of autonomic managers. For a given data center, an administrator may define the number of management levels and for each level the position of managers. Since different types of applications may come and go, there will be a dynamic number of MEs and, respectively, a dynamic number of AMs in each level. Upon arrival of any new application in the data center, the AM initiation module has to be invoked. During the AM initialization procedure, a unique ID is generated (for example, as a combination of an IP address of the host where the AM will run, the parentID or any local variables) for the AM. The AM also needs to have access to the policy repository. The policy repository server contains all policies for the AMs in the management system. The first time that an AM has access to the policy repository is at its bootstrapping phase to get initialized, although during its life cycle the AM may be asked by its parent to access the repository and get updated policies from there.

After AM initialization in which all environmental variables are initialized, the management loop starts to run

 ISBN: 978-1-61208-388-9

(see Figure 2.). The AM management loop uses the monitoring heartbeat values of its managed elements and checks for incoming messages. Messages correspond to events and a timing event happens periodically. As events occur, policies are examined and the values of the parameters are used to evaluate conditions in policies. We assume that the upper layers AMs are privileged over their child and so their policies are affected by their parent's policy. For instance, the parent can change a child AM's policy profile to "green", which could mean that the AM should give higher priority to decreasing energy consumption of its MEs than to ensuring that the SLA violations are minimized.

IV. DATA CENTER SIMULATOR

We have been developing a data center simulator [19] [28] in order to evaluate different configurations of autonomic managers and different policy sets. Our data center contains a set of systems (its definition follows) where each system runs its own kinds of applications. We can think of a data center abstractly as consisting of a number of racks $R=\{r_1, r_2, \ldots, r_R\}$ and coolers $\{c_1,c_2,\ldots,c_k\}$ laid out in some spatial configuration pattern with some network connections among them (multiple separate clusters are each collections of racks, with perhaps no communication between the racks in different clusters). Each rack is comprised of a number of chassis, and within each chassis there are numbers of servers (compute nodes).

```
Input parameters: ParentID, AMLevel, ProfilePolicy, ME, heartbeat,
        heartbeatValue, configVector, configVectorValue
1.  begin
2.    update heartbeat value
3.    While (!messageQue.isEmpty())
4.    begin
5.        msg=messageQue.dequeue
6.        if (msg.opcode== ReqForHeartbeat)
7.          send(UpdateHeartbeat, ParentID,AMID, heartbeatValue>
8.        if (msg.opcode== ChangeProfilePolicy)
9.          Update ProfilePolicy with received one in the message
10.       if (msg.opcode== PolicyChange)
11.         Update the policy received in the message with the one that
      already is in AM policy set
12.       if (msg.opcode== UpdateConfig)
13.         Update corr. param. in configVectorValue with parameter in
      the message
14.   end
15. //all triggered event are put in a queue
16.   while (!eventQu.isEmpty() )
17.   begin
18.       EV= eventQu.dequeue
19.       for (all PL∈ ProfilePolicy)
20.       begin
21.         invoke applied policy
22.       end
23.   end
24. end
```

Figure 2. Management Loop

On top of this physical infrastructure, we have defined a System; our terminology for a number of computes nodes inside a number of racks, which are capable of running the same type of jobs. We assume that systems are defined in terms of a set of racks. Compute nodes inside racks can be shared between different applications that run on that system. At any time, a node n_i inside the system is either assigned to an application/user or ready to be assigned; also, individual nodes can be powered on or off (put to sleep).

Application behavior and workload are key elements in data center operations and have a direct impact on the energy consumed by a system and hence a data center. In our model, we consider three broad classes of applications:

Interactive: This system provides access to users across the Internet/intranet, such as web servers, transactional servers, etc. These applications process short requests (transactions) and fast response time is the main objective of these types of applications. We model this as an InteractiveSystem.

Enterprise: This system provides applications to different business units, where applications may require large amounts of secure, reliable data storage and high availability, running 24/7, e.g. a human resources system. Workloads in these kinds of application vary – from short requests/jobs to much longer activities, e.g. report generation. The key characteristic is that these systems typically run for long periods.

High performance computing (HPC): This system runs scientific applications in batch mode and typically needs multiple CPUs to do high computation jobs.

With this definition in mind, we can think of a data center as a set of systems running different types of workloads, so our logical model of a data center is:

$DC=\{sys_1,sys_2,\ldots,sys_i\}$ where sys_i is a system and a system, then, is defined as:

<Name, RA, Sch, Rack-list, Node-list, App, AM>

where:

Name: is a system id.

RA: is resource allocation algorithm assigned to the system. Assigning any compute node to the application is done by this algorithm. Anytime that management polices force an application to release/ allocate a compute node this algorithm will decide.

Sch: is a scheduling algorithm for all applications running in the system. It has just one output which is next job (any type) to be run.

Rack-list: a list of racks assigned to this system.

Node-list: a list of compute nodes that are assigned to this system. There may be situations which a rack is shared between a number of system. In this case list of each system compute node is important.

App: specifies the type of applications that run on the system; Enterprise (Ent) applications, Interactive Applications (Int) or HPC applications.

AM: autonomic manager attached to this system.

The applications that run on a particular system are described as follows:

- *Ent*: An Enterprise system has a number of applications, each application having an interactive type workload running on a list of servers and its own SLA violation description.
- *Int*: An Interactive system deals with a list of dynamic coming-going workload from users. This type of

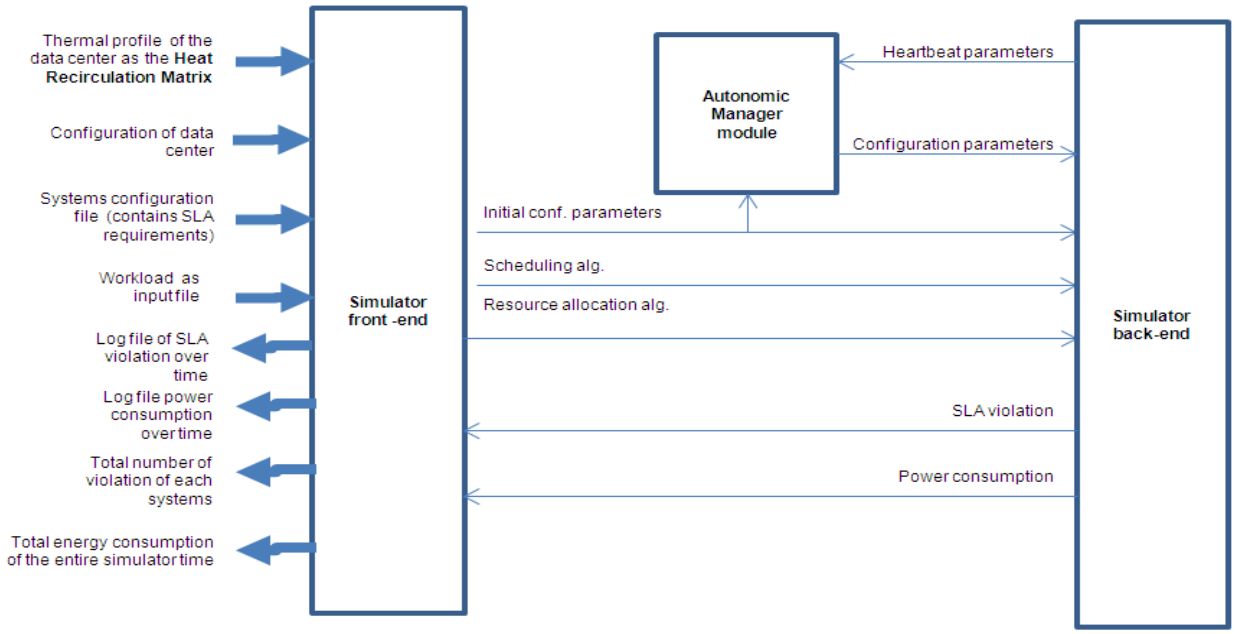


Figure 3. Overall Structure of Simulator

workload has arrival time, duration, and SLA violation definition. They are web based type applications.

- *HPC*: An HPC system just has HPC type jobs; each job has a duration, deadline, needed CPU utilization and number of nodes (for parallel processing jobs).

Putting this information together, a data center is then defined as:

<*RackList, Cooler, SysList, RedTemp, ThermalMap, AM*>

where:

RackList: is list of racks in the data center; information regarding chassis and blade servers inside the rack is part of the rack definition.

Cooler: is the cooling specification. The efficiency of the Computer Room Air Conditioner (CRAC) depends on air flow velocity and conductivity of materials which is quantified as the Coefficient of Performance (COP).

SysList: is the list of defined systems.

RedTemp: Red temperature: the maximum temperature that hardware in the data center can tolerate; this parameter will affect the cooling energy consumption.

AM: is the manager of the whole data center.

thermalMap: is used to calculate energy consumption of the data center; the thermal model used in this research was developed by Arizona State university [20]. Briefly, the computing and cooling power in data center are considered where the thermal model is a matrix, where an entry in the matrix specifies how much generated heat from each server will re-circulate to other servers. The overall structure of our simulator is presented in Figure 3. The illustrated autonomic management module is in charge of coordination and planning among different AMs across the data center. The simulator has been evaluated with different types of systems.

V. EXPERIMENTS

To illustrate the impact of our proposed management

TABLE II. SLA PROFILE FOR AM ATTACHED TO HPC SYSTEM/WEBSERVER

PL0:
On Event: **Timer Triggered**
If (SLA is violated)
begin
(Increase freq. of all busy nodes → Activate all sleep nodes
End

TABLE III. GREEN PROFILE POLICY FOR AM ATTACHED TO HPC SYSTEM/WEBSERVER

PL1:
On Event: **Timer Triggered**
If (SLA is violated)
begin
(Increase freq. of just fully utilized CPU node → Activate just half of sleep nodes) | Activate just half of sleep nodes
end
PL2:
On Event: **Timer Triggered**
If (SLA is not violated)
begin
(Decrease freq. of all nodes → If node is ready and is not used make it sleep) | If node is ready and is not used make it sleep
End

TABLE IV. AM ATTACHED TO COOLER

PL3:
If (Max temperature is greater than Red temperature)
begin
Send UpdateHeartbeat message to AM in DC level
End

 ISBN: 978-1-61208-388-9

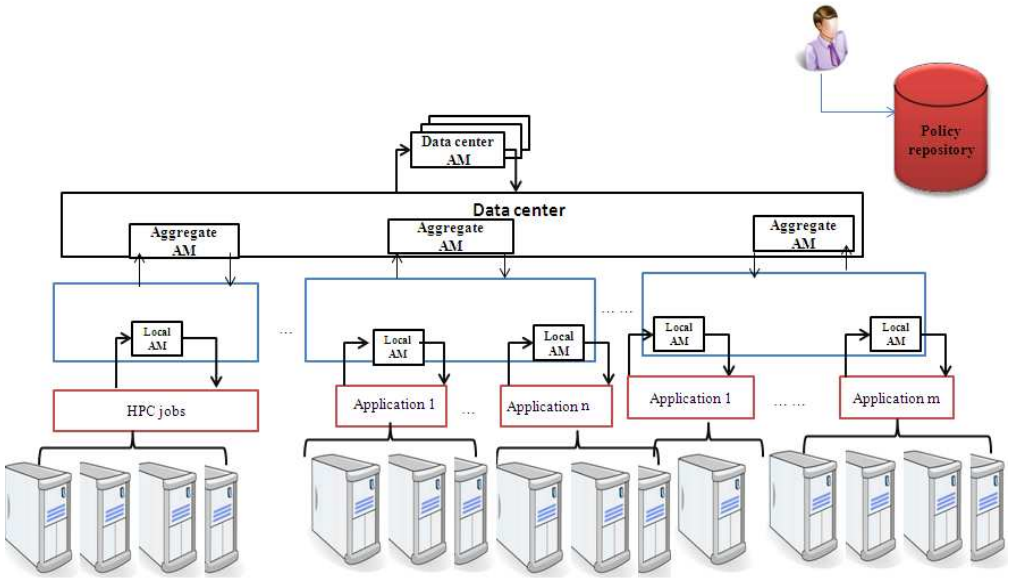


Figure 4. Prototype Management System

system, we evaluate different management scenarios for a hypothetical data center with and without management system. We describe two sets of experiments. In the first set we look at the impact of policies in managing the behavior of a hosted webserver in the simulated data center. In the second experiment, we present our prototype management system. We compare the effects of different policy profiles on energy consumption and SLA violations. For this experiment, we consider a data center with one or two HPC systems.

A. *Experiment 1: Webserver Management*

Our simulated data center has 10 racks and in each rack there are 5 chasses, each has 5 servers (in total our data center has 250 compute nodes). The simulated servers are Proliant HP DL320. This server has a standby power consumption of 5Watts; when it is idle it consumes 100 Watts and, with a fully utilized CPU, it consumes 300 Watts [24][25]. The DL320 has an Intel® Xeon® E3-1200v2 processor which has frequency scaling levels which are 3.07, 3.2, and 4.2 GHz, which when normalized to the "base level" are 1, 1.07 and 1.37. The simulated data center has one HP cooler (refer to the thermal model in IV).

In this experiment, we host a webserver in the data center with 80 (minimum) to 90 (maximum) compute nodes allocated to the webserver. Compute nodes are allocated to the webserver. Workload is scaled up version of traffic from 1999 world cup web traffic. We have attached a manager to the webserver which monitors the SLA and, according to its active polices, does some actions. SLA is violated when the response time is more than two simulator cycles. Two distinct sets of policies are considered: a green profile and SLA profile (see Table II. and Table III.). These policies are trying to minimize energy consumption (Green policy profile) and minimize SLA violations (SLA policy profile). The SLA policy profile is a time-triggered policy (every 60 seconds). When triggered, the AM checks for any SLA violations in the system and tries to do dynamic CPU frequency scaling and activate sleep allocated compute nodes. If frequency scaling is not supported by compute nodes, this policy just activates sleeping nodes. The simulator counts violations during policy timer period. Green policy profile also tries to do dynamic frequency scaling and activation/deactivation of compute nodes if SLA violations happen. This profile tries to keep active compute nodes and CPUs at moderate frequency levels based on whether there are SLA violations or not. Results show (Table V.) that the Green policies result in less consumption of power than SLA based policies. In SLA based polices; the total energy consumption (cooling and computing) is not available since the inlet temperature is exceeded 475 times and this is not handled in the simulator. The main objective of this experiment is to show the scalability of the developed data center and also the impact of policy based management.

B. *Prototype Management Environment*

We have modeled our prototype management system (illustrated in Figure 4.), using a three level hierarchy. At the bottom level, we have local AMs. Each local AM is attached to a number of compute nodes. The second level of AMs are called aggregate AMs; they logically aggregate management responsibility from the local level to the data center level. AMs at this level have the AMs at the first

TABLE V. COMPARISON BETWEEN GREEN AND SLA PROFILE POLICY IN SEMI-LARGE SIMULATED DATA CENTER

Scenario	Green	SLA
Computing power of Webserver	7.7 * 10^8	9.6*10^8
total energy consumption (Watt × Simulation Time)	1.9*10^9	N/A
Mean power consumption (Watts)	26982	N/A
Number of times crossing red temperature	0	475

level as their managed elements. At the top level, there is one (or more for replication) Data Center AM which is the coordinator among all aggregate AMs. AMs in our management system cooperate and so exchange information with the other AMs in the level above or below. This information is called heartbeat and configVector data that is the sensor and actuator information. Heartbeat information can be fetched by the parent periodically or upon the occurrence of any event, i.e. specified in the event part of AM policy set, e.g. on an SLA violation or power cap violation. Aggregate level or data center level AMs may inquire of their children for heartbeat updates to make better decisions. Any changes in configuration parameters or policies of the child are then sent from the parent AM as configuration parameters. We have simulated this prototype for just HPC type workload. We have attached an AM (local AM) to a HPC system and for number of HPC system (in our case we have two) we have a data center level AM which manages HPC systems behavior beside other AMs (the AM attached to the cooler in following it will be explained) in the data center.

C. Experiment 2: HPC Data Center

In this experiment, we assume that each chassis has one blade server, so, in total, the data center has 50 physical servers configured into two separate HPC systems; one of 30 compute nodes and one of 20 compute nodes. Each of our HPC systems runs an HPC workload consisting of long and short batch jobs (their workloads are not the same). Each job in the workload has an arrival time, duration, needed CPU utilization (will be used for thermal model) and deadline (maximum waiting time in the system before dispatching to a compute node). An SLA violation occurs when a deadline is passed for a job in the workload. Two system workloads have 730 and 173 jobs respectively, which on average demand 3 compute nodes [27].

Ponder-like [21] notation has been used to describe our policies. For the simulation study, we assume, we have two policy profiles at the system level: a Green and SLA policy profiles as per our webserver experiment. We also have a data center AM which can change the system level AM's policy profile based on their SLA violation status. If there is any SLA violation on any of the HPC systems, the policy tries to change the policy profile of all systems based on whether they have an SLA violation (change it to SLA) or not (change to Green) (see policies PL4 and PL5 in Table VI). Upon any SLA violation in any system, they will send their heartbeat (SLA violation) to the data center AM. Data center AM will evaluate its policies and change any system with a violation to an SLA based policy and the rest of the systems will be set to Green. We expect that, by changing policy profile of system dynamically, we can get better results in terms of total energy consumption and still limit the number of SLA violations. Data center AM has policy to deal with the Cooler - if the AM in the cooler detects a red temperature then it sends message to the data center AM (see Table IV.). There is a policy for the data center AM to "block" an HPC system with lowest priority for a period of time (timer is set to 2 minutes) to relief data center load. What we have simulated for blocking HPC system is not running any jobs from the workload and in case of new arrival jobs just queuing them and not dispatching them to the compute nodes.

D. Experimental Scenarios

Five different scenarios have been considered to evaluate the performance of having multiple autonomic managers with varying sets of policies.

Scenario 1. No management: The data center has the two running HPC systems, one with 30 compute nodes and a workload of 730 jobs and another system with 20 compute nodes and a workload with 173 jobs.

Scenario 2. There is a manager at the system level, which has a *SLA* policy profile (see Table II.SLA Profile For AM Attached to HPC System/Webserver). This scenario runs for the small HPC system of 20 compute nodes (we assume that the large HPC system is not running

TABLE VI. GREEN POLICY PROFILE FOR DC LEVEL AM

PL4:
On Event: (**SLAViolation$_1$ | SLAViolation$_2$**)
if (SLA$_1$ is violated)
begin
Switch system$_1$ to SLA based
end
else
begin
Switch system$_1$ to Green
end
	PL3:
	On Event: **"block timer" trigger**
	if (true)
	begin
	Unblock the blocked system
	End
	
	PL5:
	On Event: (**SLAViolation$_1$ | SLAViolation$_2$**)
	if (SLA$_2$ is violated)
	begin
	Switch system$_2$ to SLA based
	end
	else
	begin
	Switch system$_2$ to Green
	end

PL6:
On Event: **Receiving UpdateHeartbeat from Cooler**
if (true)
begin
(block HPC system with lowest priority ->start a timer: "block timer")| Switch strategy of all others to Green
end	

TABLE VII. COMPARISON BETWEEN DIFFERENT SCENARIOS

Scenario	No management system (Scenario 1)	Single AM in system level (Scenario 2 and 3)		Multiple AMs (data center level and system level) (Scenario 4 and 5)	
Profile Policy	N/A	Green	SLA	DC AM Profile policy is Green	
Num. of HPC systems in DC	2	1		1	2
Number of SLA Violation	448	189	187	189	454
total energy consumption (Watt × Simulation Time)	N/A	8,000,000	9,800,000	8,318,000	23,171,143
Mean power consumption (Watt)	N/A	6,430	7,287	6,518	11,956
Number of time crossing red temperature	91	0	0	0	3
Number of exchanged messages	0	226	22	253	2,764

in the data center). The goal here is to evaluate the impact of the *SLA* policy profile on power and performance.

Scenario 3. Exactly as scenario 2 except the *Green* policy profile for system level AM.

Scenario 4. We have two managers: one AM at the system level (the small HPC system is running) and data center level. The data center AM's policy profile is *Green* (see Table VI.). We aim to evaluate the impact of changing policy profiles of the HPC systems dynamically on the power and performance. The main goal in Scenario 4 is to consider how the data center level AM impacts the behavior of its lower level AMs. While data center level AM is green means that it makes the system level AM to behave close to when it is Green itself.

Scenario 5. We have both HPC systems with their AMs running, an AM at the data center level, and the cooler has its own manager that just checks for its maximum inlet temperature. If the inlet temperature is greater than the red temperature of hardware in the data center (specified in the data center configuration), the cooler AM sends a message (UpdateHeartbeat message) to the data center AM asking it to do something (refer to Table IV.). In this scenario, we assume that the policy available to the AM in the data center indicates that a system should be "blocked" – essentially decrease processing by not executing additional jobs. Obviously, the blocked system will suffer from SLA violations but the gain is that this decision addresses exceeding the red temperature for the whole data center. System priority is defined with the HPC system configuration.

E. Experimental Results

The result of running these scenarios is shown in Table VII. The first scenario does not have a management module and has two HPC systems. Running these systems under the workloads results in the inlet temperature of the cooler exceeding the red temperature 91 times; as a result, the simulator is not able to calculate the total energy consumed. As shown in Table VII. , Scenarios 2 and 3 involve a single HPC system with a manager. The *Green* policy profile consumes less energy and power than the same HPC system with the *SLA* policy profile while the number of SLA violations is about the same. This scenario shows how a small difference in policies can affect the overall behavior. In Scenario 4, we consider an AM at the data center level and its policy profile is *Green*. The data center AM with the *Green* policy profile is configured to dynamically change the policy profile of system level AMs in accordance with the system's SLA violations; if there are SLA violations at the system level, its policy profile is altered to be *SLA* based in order to put more priority on achieving SLAs than on energy conservation. The result shows that by having a data center level manager able to dynamically switch its corresponding system level AMs profile we can get the same results as when the system level has Green profile policy. In Scenario 4, it is data center level AM that controls the behavior of the system level AMs and by setting data center to Green we implicitly mak system level AM to behave close to Green profile policy.

F. Management Overhead

The management system is responsible for configuring managing entities and making sure they have updated context information. All communications are based on message passing, which causes network traffic. Although hierarchical architecture is expected to have less communication overhead, we consider the number of exchanged messages between managing entities as management overhead. As shown in Table VII. , the last scenario which has an AM attached to the cooler and dual levels of AM is expected to have more messages. These messages are passed between four different zones: within the two HPC systems, between the data center AM and the HPC system AMs, and between the cooler AM and the AM in the data center. The other expected overhead is due to the actual computing resources consumed for management activities (from the initialization of the management system to running the MAPE loop in each manager). We do not actually execute managers within the simulation, so we cannot get an estimate from the simulator itself. However, to estimate computing resources consumed by a manager, we ran a "pseudo-manager" that executed and timed a MAPE loop with 10 policies with fairly CPU intensive actions as well as accessing a file to simulate the reading of policies – something that would not normally happen each time through the MAPE loop. This manager was run on a computer of roughly the same computational power as the HP DL320. The result showed that this MAPE loop consumes 0.00002% CPU utilization of the processor which is essentially negligible. Even if our measure is off by a

 ISBN: 978-1-61208-388-9

factor of a thousand, the management overhead to evaluate policies and initiate actions is small. Of course, the resources consumed in executing those actions could be substantial, e.g. a virtual machine migration, but this depends on the specific actions and what makes sense in the contact of managing the system.

VI. CONCLUSION AND FUTRUE WORK

This research aims to develop an autonomic management system that can help reduce data center energy consumption while still adhering to service level agreements and performance expectations. The results of combining autonomic computing and policy based management suggest a useful approach. We considered a hierarchical arrangement of autonomic managers that is based on the physical position of managed elements. A general approach for an autonomic management system has been introduced. The core principles that drive the management model are: a message passing approach and policy-driving autonomous managers. The approach has been evaluated on a hypothetical data center using a simulator. The simulator in this experiment has 50 nodes but has the ability to be extended by increasing the number of blade servers in each chassis. (Here we have one blade server in each chassis; webserver experiment addresses scalability). The results show that, first, by having simple policies (such as SLA and Green policies), we obtain an acceptable reduction in power consumption. The second lesson learned is the impact of upper layer AM profile on the behavior of its lower level AM. As shown in scenario 4 upper layer AM being Green causes the same behavior as while the system level AM is Green. Comparison of Scenario 1 and 5 shows that the proposed three-level management hierarchy with given policies controls the behavior of the data center in terms of minimizing power consumption with negligible management overhead and effect on SLA violation. The result of having different profile policy and different level of management shown in this work are workload agnostic. We have run experiments with web-based workload that shows promising horizon for our management system. Future work will look at the management algorithms and dealing with changes in the computing environment, e.g. the dynamic start of or termination of applications. It will also explore means for more general cooperation between managers and for different configurations for the management system, e.g. peer-to-peer, etc.

REFERENCES

[1] J. Koomey, Growth in data center electricity use 2005 to 2010. Analytics Press, August 2011.

[2] B. Anton, J. Abawajy, and R. Buyya. "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing." Future Generation Computer Systems 28.5 (2012): 755-768.

[3] W. Forrest, J.M. Kaplan, and N. Kindler, “Data centers: How to cut carbon emissions and costs,” mckinseyquarterly.com, November 2008.

[4] Google green, http://www.google.com/green Last visited July 2014.

[5] Kh. Bithika, S.Hariri, and M.S. Yousif. "Autonomic power and performance management for computing systems." Cluster computing 11.2 (2008): pp.167-181.

[6] Computing, Autonomic. "An architectural blueprint for autonomic computing." IBM White Paper (2006).

[7] J. D. Moore, J.S. Chase, P. Ranganathan, and R.K. Sharma. "Making Scheduling" Cool": Temperature-Aware Workload Placement in Data Centers." In USENIX annual technical conference, General Track, pp. 61-75. 2005.

[8] http://institutes.lanl.gov/hec-fsio/ Last visited July 2014.

[9] K. Catriona. "Decentralised Metacognition in Context-Aware Autonomic Systems: Some Key Challenges." In Metacognition for Robust Social Systems. 2010.

[10] R. Boutaba, and A. Issam. "Policy-based management: A historical perspective." Journal of Network and Systems Management 15.4 (2007): pp.447-480.

[11] J.O.Kephart, and D. M. Chess. "The vision of autonomic computing." Computer 36.1 (2003):pp. 41-50.

[12] T. Mukherjee et al. "Spatio-temporal thermal-aware job scheduling to minimize energy consumption in virtualized heterogeneous data centers." Computer Networks 53, no. 17 (2009): pp.2888-2904.

[13] T. Mukherjee, et al. "Model-driven coordinated management of data centers." Computer Networks 54.16 (2010): pp.2869-2886.

[14] J.O.Kephart. "Coordinating Multiple Autonomic Managers to Achieve Specified Power-Performance Tradeoffs." Fourth International Conference on Autonomic Computing, IEEE Computer Society, 2007.pp. 24-33.

[15] Z.Abbasi. et al. "Thermal aware server provisioning and workload distribution for internet data centers." In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing,ACM 2010, pp. 130-141.

[16] D. Kusic, J.O. Kephart, J.E. Hanson, N. Kandasamy, G. Jiang, “Power and performance management of virtualized computing environments via lookahead control”, Cluster Comput. 12 (1) (2009) pp. 1–15.

[17] J.D.Baldassari, et al. "Autonomic cluster management system (ACMS): A demonstration of autonomic principles at work." Engineering of Computer-Based Systems, 2005. ECBS'05. 12th IEEE International Conference and Workshops on the. IEEE, 2005. pp. 512-518.

[18] R.Bahatl, M.A. Bauer, Elvis M. Vieira, and O. K. Baek. "Using policies to drive autonomic management." In Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks, IEEE Computer Society, 2006, pp. 475-479.

[19] http://publish.uwo.ca/~fnorouz/publication/Simulator.pdf. Last visited July 2014.

[20] http://impact.asu.edu/BlueTool/wiki/index.php/BlueSim. Last visited July 2014.

[21] http://www.ponder2.net/. Last visited July 2014.

[22] R. Thomas, et al. "Autonomic management of large clusters and their integration into the grid." Journal of Grid computing 2.3 (2004): pp.247-260.

[23] http://eu-datagrid.web.cern.ch/eu-datagrid/ Last visited July 2014.

[24] T. Gerald, et al. "A multi-agent systems approach to autonomic computing." In Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1, IEEE Computer Society, 2004. pp. 464-471.

[25] Standard Performance Evaluation Corporation. http://www.spec.org/power_ssj2008/results/power_ssj2008.html. Last visited July 2014.

[26] A. Richard, M. Pelc, and H. Shuaib. "The interoperability challenge for autonomic computing." EMERGING 2011, The Third International Conference on Emerging Network Intelligence. 2011.

[27] http://institutes.lanl.gov/hec-fsio/ Last visited July 2014.

[28] https://github.com/fnorouz/simulator/ Last visited January 2015.

 ISBN: 978-1-61208-388-9

Experimental Validation as Support in the Migration from SQL Databases to NoSQL Databases

Abraham Gomez
Département de génie logiciel et des TI
École de Technologie Supérieure (ÉTS)
Montreal, Quebec, Canadá
abraham-segundo.gomez.1@ens.etsmtl.ca

Rafik Ouanouki
Département de génie logiciel et des TI
École de Technologie Supérieure (ÉTS)
Montreal, Quebec, Canadá
rafik.ouanouki.1@ens.etsmtl.ca

Anderson Ravanello
Département de génie logiciel et des TI
École de Technologie Supérieure (ÉTS)
Montreal, Quebec, Canadá
ravanello@gmail.com

Alain April
Département de génie logiciel et des TI
École de Technologie Supérieure (ÉTS)
Montreal, Quebec, Canadá
alain.april@etsmtl.ca

Alain Abran
Département de génie logiciel et des TI
École de Technologie Supérieure (ÉTS)
Montreal, Quebec, Canadá
alain.abran@etsmtl.ca

***Abstract*—NoSQL databases, also known Not only SQL databases, is a new type of databases that provides structures other than the tabular relations used in relational databases, for storage and retrieval data. This new databases are now a valuable asset to design complex real-time applications that use Big Data in cloud environments (NoSQL cloud databases). Today, the migration process from relational databases to NoSQL databases is unclear and mainly based on heuristics approaches, such as the developers' experience or intuitive judgments. This paper which forms part of a more extensive research project regarding how the design and use of a guidelines set could improve the migration process. The results present an experiment designed to obtain a baseline that allows an effective comparison between two migration processes: the first one, without the use of any guidelines and based on the traditional heuristic approach and the second one, with the guidelines. The experiment reports that the use of such guidelines improves the migration process.**

Keywords-Column oriented databases; NoSQL databases; distributed databases; software experimentation; cloud computing.

I. INTRODUCTION

Cloud Computing (CC) is rapidly becoming an integral part of our daily life. The applications developed for this new way of computing has brought new challenges for software engineering because of the large amount of data that is collected by the CC software applications. In fact, these applications accumulate and analyze a lot of information on a daily basis, which has led to create a new research area called "*big data*".

Until a few years, companies had used massively the relational database technology (RDBMS) to deal with this bigdata. However, Abadi [2] states in his research that accessing petabytes of data efficiently using RDBMS, in the cloud, is very challenging, and solutions like sharding, creates many other problems. At this point, the NoSQL databases emerge as a solution to these challenges.

This generates the problem about how to migrate the data from RDBMS to NoSQL environment. Unfortunately, little work has been done to explore the migration from RDBMS to NoSQL. As a matter of fact, there have only been preliminary researches focusing on certain minimum elements such as tables or types of relationships. One first approach was reported by Chonxing [3], who proposed some migration rules for a conversion to HBase. However, more experimentation is needed, to show that a one-size-fits-all approach is not possible and, more importantly, not all applications are good candidates for this migration, according to Stonebraker in [4] [6].

Despite the above situation, there are a lot of enterprises that offer the migration service from RDBMS to NoSQL, but they conduct this migration using a heuristic approach that implies a lot of experience in NoSQL environments. On the contrary, the use of a standardized formal way could help those that are NoSQL neophytes, but have experience in RDBMS.

This paper forms part of a more extensive research project regarding how the design and use of a guideline set could improve the migration process from RDBMS to NoSQL. At the end of the project, the idea is to offer something that really improves the process; so, it is important to have some baseline that allows, at least, a

 ISBN: 978-1-61208-388-9

comparison. The work presented here focuses on the design of an experiment to establish a baseline that allows a valid comparison between the migration process of a database from RDBMS to NoSQL. This paper reports on the design of the experiment, its application and its results. In a near future paper, a second group of RDBMS experts will conduct the same migration, but using guidelines developed by our research team.

The paper is organized as follows. Section II presents the problem statement. The research objectives are presented in Section III, followed by the Section IV with the related and previous work of the project. Section V presents the experimental design and Section VI reports the case study results. Section VII presents the conclusion, and finally, the future work is presented in Section VIII.

II. PROBLEM STATEMENT

Although, it is still very popular to use the relational database model for CC applications, when the data deployed in the database servers (in the cloud) grow beyond 1TB, this technology starts to show its limits, e.g., in their work Stonebraker [4][6][8] state the volume of data stored is related with problems in response time in the research field of "big data".

Also, the large increase in the number of users connected to cloud applications can cause other problems, such as transactional difficulties, storing space management and the non-compliance of ACID properties (ACID is an acronym for Atomicity, Consistency, Isolation, and Durability. Complying with all these properties guarantee that database transactions are processed reliably). That is, the administration of these systems becomes more and more complex, as reported by Abadi [2].

When a big data application, using a relational database technology, reaches its limit and the solutions, such as, sharding, are failing to solve the issues its time to think about NoSQL technologies. This solution provides new levels of economies of scale, agility, and flexibility compared with traditional IT environments based on the relational database model.

The problem that the whole project wants to address is as follows:

Since the industry uses mainly relational databases and they are likely to migrate some of their large scale existing applications to a NoSQL model, there is a research need to improve this process by identifying a set of guidelines to help database specialists in this first time migration from RDBMS to NoSQL database. Our experiment will be focusing on an HBase migration, which is a popular column-oriented NoSQL database developed as part of Apache Hadoop project.

III. RESEARCH OBJECTIVES

First of all, it is necessary to make a distinction between the research objective and the paper's scope. The research objectives is the design and use a set of guidelines as a way of improving the migration process of databases from RDBMS to NoSQL databases, focusing on HBase. In this article, the concept "improve" is going to be used in the sense to bring into a more desirable or excellent condition the current migration process from SQL database applications to NoSQL database applications. Fig. 1 shows, graphically, the difference between the two things (the paper scope is inside the red dotted border). The idea will be use the same relational database application and follow two experimental tracks: the first, without the use of the guidelines, and the other one with the use of the guidelines. At the end of the project, two HBase databases will be obtained and compared (see Fig. 1).

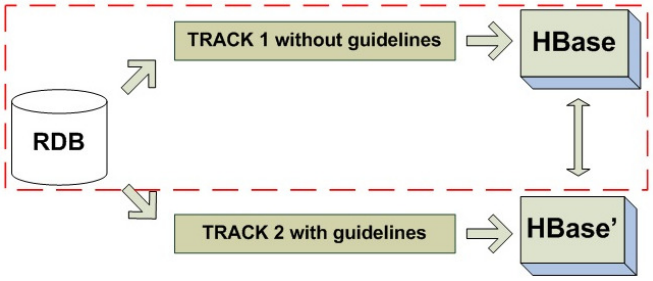


Figure 1. This paper scope versus the overall research objective.

In order to conduct a comparison between the applications and resulting databases HBase and HBase', some preliminary experiments must be conducted. The scope of this paper is to create a baseline that allows a valid comparison between the migration process of a database from RDBMS database to NoSQL database (HBase), without the use of guidelines (heuristic approach) and with the use of guidelines (the proposed solution)

IV. RELATED WORK

Few researchers have addressed the problem of migration of a software application from RDBMS environment to NoSQL. Indeed, there is very little literature on how doing it, at least in a standard way; this fact is supported in the references, because there is no recent research work on this field, the closest is around 2011, and the others references are significantly older than that.

The options currently available to accomplish this migration are mostly based on a heuristic approach. It means, based on the developers' experience, educated guess, intuitive judgment, or common sense. This approach does not guarantee that an optimal solution will be found; but, if it is properly done, it can provide a satisfactory solution.

Salmen has proposed, in his initial attempt focused on identifying some of the core activities that are common to every migration process, to draw some general conclusions about how start this migration process (e.g., the DDI methodology proposed by Salmen [13]), where DDI stands for Denormalization, Duplication, and Intelligent keys.

Denormalization is the process used to optimize the read performance of a database by adding redundant data or by regrouping data. Data duplication can be defined as the occurrence of redundant data within a system by various means. An intelligent key is a database key which depends wholly on one or more other columns in the same table. An intelligent key might be identified for implementation convenience, when there is no good candidate key.

 ISBN: 978-1-61208-388-9

Another contribution to this research field was published by Chongxin who developed some rules to help in the migration from relational databases to a specific cloud computing database, which is HBase [3]. However, Chongxin explored a reduced set of ideas that he called "rules" (three to be precise) and these rules do not cover the entire characteristic that implies a relational database application today. Besides, Chongxin establishes their rules in a consecutive order, it means, in the first stage, one must apply the rule number 1; then, the rule number 2, and finally, the rule number three. This way of working reduces drastically the results of Chongxin's method because the solution offered only covered the relational aspect "relationships" (including the migration of "one-to-one", "one-to-many" and "many-to-many" relationships), the author does not offer a way to deal with other relational aspects like tables, fields, store procedures or triggers. Furthermore, the rules were designed to be applied in a pipeline way (one after the other) and not as the user needs it, indeed, the Chongxin's work does not offer any method to deal with the case if one user tries to apply the rule number two, without applying the rule number 1 previously.

Notwithstanding the above, the Chongxin's work offers an initial approach to the problem of migration a RDBMS to NoSQL. In fact, separate the relationships into its different types and create a rule for every type is an excellent first solutions' approach.

The last work was suggested by Singh in [14][15]. In their work some general guidelines were proposed, but the problem is the guidelines were developed using the methodology of use cases that follow a heuristic approach and reduce the possibilities to replicate the work or adapted in general ways to applied in other contexts.

It is clear from the above that there is a twin problem: first and foremost, a new perspective to address the migration problem is needed. But, along with this new solution, a way to measure the impact of the new solution is also necessary. This document is part of an entire research whose goal offers some guidance for converting an RDBMS to NoSQL database based on a guideline set. The goal of the paper is to offer an experimental baseline that allows the comparison results between two migration's processes, one using the guidelines and another one without the use of the guidelines.

V. EXPERIMENTAL DESIGN

As stated earlier, in Section III, this experiment is the first part of the entire research project to address the migration problem from relational to NoSQL databases. Only the results of the experimentation of track 1 will be presented and used as a baseline for the future comparison with the track 2 experimentation results (see Fig. 1). This experimental design was based on Jedlitschka's work [16].

A. *Hypotheses*

The null hypothesis is:

H_0: "there is no real improvement in the migration process with the use of guidelines; if there is any advantage, it is only coincidental, and the best option is to use a heuristic approach based on the developer's experience".

On the other hand, the alternative hypothesis is:

H_1: "there is a significant improvement in the migration process with the use of guidelines; this is not coincidental and the better option to achieve this process is to use the guidelines".

B. *Case study participants and data collection procedure*

This subsection follows the structure of Easterbrooke [19], Marcos [20], and Zelkowitz [1]; moreover, the experiment was designed using the point of view of a typical developer. Taking this point of view, the participants were asked to state their experience level and they were classified according to 1) their academic background 2) working field; 3) number of years of work experience with relational database, and 4) the number of years of work experience with any NoSQL database.

The word "*experience*" was related to the domains of programmer, relational database programmer or relational database administrator.

Moreover, the classification was summarized according to different options. The academic background had the options Graduate with PhD, Graduate with Master, Graduate, and Undergraduate Student. The working field had the options Industry, Academic, and Research Center.

The number of years of work experience with relational database environment had the options of No Experience, Low Experience (less than a year), Middle Experience (2 to 5 Years), and Advanced Experience (more than 5 Years).

The number of years of work experience related to any NoSQL database had the same options as above.

The goal was obtain a classification for the participants according their experience that would allow us to know the combinations (pair) "*relational-NoSQL*" experience that needs the solution and where it can be most useful. Fig. 2, for instance, highlights the pair Low-Medium, meaning a "*low*" experience in relational database environment and "*medium*" experience in NoSQL database.

Eighteen individuals participated in the experiment: twelve participants belong to the industrial sector and four participants were graduate students at the École de Technologie Superieure (ÉTS). All participants were provided with a clear and well established knowledge about the purpose of the experiment.

Relational Database	No-SQL Database
Low	Low
Medium	Medium
High	High

Figure 2. Classification for the participants according their experience.

The material used in the execution was:

- The document including the call for participants (date, time, place and activities that took place in the workshop), which was an invitation sent by email

and telephone calls two months before the workshop.

- The participant's instructions.
- The synthetic relational schema (Blue document). This was the schema that they must have migrated to NoSQL. The schema was based on the research of Singh [14] and it was composed by seven tables, four large tables (City, Department, Doctor and Hospital) and three junction tables (DoctorDeparment, HospitalCity and HospitalDepartment,). The tables City, Department, Doctor and Hospital remain the classical "Id, Name" structure, with "Id" as primary key. Table "Doctor" contains "Id, Name, Age, Sex and BorIn". The latter field is the Id of the city where the doctor was born. The junction tables allow expressing the "many-to-many" relationship indicated by each junction table's title (see Fig. 3). It is important to note, the participants were offered the opportunity to choose between several sub-schemas from the main schema. For instance, one participant could choose only migrate the sub-schema composed by the entities Hospital – HospitalDepartment – Deparment or the sub-schema Doctor – DoctorDepartment – Deparment or the participant could select the entire schema (see Fig. 3).
- The NoSQL solution (Green document). This was an empty sheet, where the participant could draw the new schema resulting from their knowledge.
- The participants training document (White document). It was a document that summarizes the training part explained at the beginning of the experiment, including the relational database and Not-SQL explanations.
- The drafts documents (Yellow documents). It means sheets to draw any thing the participant could use as support.

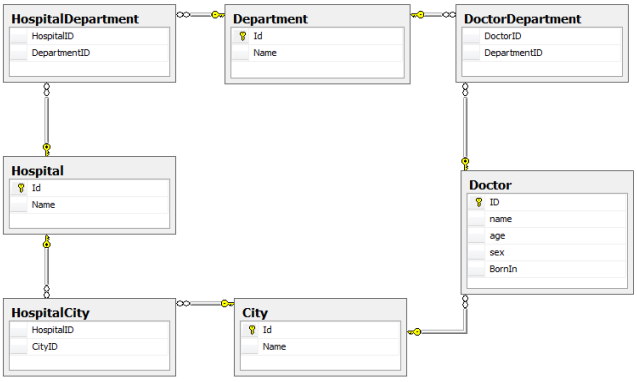


Figure 3. Relational schema given to the participants.

- The final survey form, which was applied to the participants, after the experiment. Besides, it was one of the two measurement instruments used in the experiment. The second one was the schema designed in the green document. The survey was designed following the research work devised by Kasunic [21] and Lethbridge [22]. It was composed of nine questions, with the first four were totally oriented to "experience classification", as explained earlier. The fifth question was related with the migration process and the opinion about the first step to begin it. The sixth question was related with the effort needed to achieve the process without the guidelines. This question was rated from 1 to 5, where 1 indicates that the process was easy to achieve without effort, a value of 3 indicates that it was required a maximum effort to achieve it and a value of 5 means that no matter how comprehensive the effort, it was not possible to achieve it. The seventh question was designed to evaluate their level of confusion during the process, e.g., no idea where to start or what the next step was. The questions were rated from 1 to 5, where: always confused, very often confused, sometimes confused, rarely confused, and never confused. The eighth question is a matrix for evaluating the percentage that covers the designed solution with regards to the relational aspects mentioned earlier (Table, Constraint, PK, and FK). Finally, the ninth question, the participant's opinion to know if he/she thinks that to receive some guidelines could improve their process. This question was rated 1 to 5 with the levels: strongly agree, agree, undecided, disagree, and strongly disagree.

VI. CASE STUDY RESULTS

As previously mentioned, there are no experiments and data that support conclusions or decisions in the domain of migration from RDBMS to NoSQL databases. Generally speaking, all the migrations have been conducted using a heuristic approach, e.g., the developers experience or the developer's educated guesses or their common sense.

The goal of this paper is using the results obtained as a baseline for comparisons in future stages of the entire research. The experiment process consisted of two well established parts, first at all an explanation of all the technological context, it means, a tutorial about the RDBMS and the NOSQL technology, a duration of 30 minutes was scheduled. After, all the participants received the documentation stated in the section above. Subsequently the participants conduct the experiment, eventually filling the green sheet (the NoSQL schema resulting from the migration). Finally they expressed their opinions filling a survey.

Table I indicates the different educational level of the participants. Generally speaking, it reports a low level of interest from undergraduate students to participate in this kind of studies. Besides, in the participants is found an 89% of graduate that shows an interest to conduct the experiment. (50% graduates with master plus 39% of graduates).

It can be observed in Table III that a great number of participants have the experience in RDBMS field; 45% have more than 5 years of experience and this result together with Table II's result (83% of participants in industry sector) give a lot of value to the results of this experimentation.

 ISBN: 978-1-61208-388-9

TABLE I. EDUCATIONAL LEVEL OF THE PARTICIPANTS

Educational Level	
Classification	***Response in percentage***
Graduate with Phd	5%
Graduate with Master	50%
Graduate	39%
Undergraduate	6%

Table II shows a great participation from industry sector (83%).

TABLE II. WORK AREA OF THE PARTICIPANTS

Area of work	
Classification	***Response in percentage***
Industry	83%
Academic	17%
Research Center	0%

In contrast, Table III also shows that 94% of the participants have no knowledge about NoSQL databases technology. The results shown by Table III strongly indicate that a set of guidelines could be an invaluable tool for the RDBMS experts in migration process.

TABLE III. TABLE LEVEL OF EXPERIENCE IN DB

Type of DB	**Experience in years**			
	No Exp	***Low Exp (< 1 Year)***	***Middle Exp (2-5 Years)***	***Advanced Exp (>5 Years)***
RDBMS	22%	11%	22%	45%
NoSQL	94%	0%	6%	0%

With regards to the first thing to do at the beginning of migration process, Fig. 4 provides the different paths presented in the participants. Considering Table II (83% in industry sector) and Table III (45% with more than 5 years of experience), there was a large proportion of 61% (resulting from 33% plus 28%) of the participants that chosen to begin with the RDBMS "tables" element (see Fig. 4). This leads to think that start by the tables could be a good guess.

The difficulty during the whole process is reported by Fig. 5. As can be seen, the initial perception that the procedure is difficult was unchanged (near 78% resulting from 39% plus 39%). This notion was reinforced considering also Table III data, about experience in NoSQL databases. So, the participants think the process demands a considerable amount of effort, because the NoSQL databases are totally new for them.

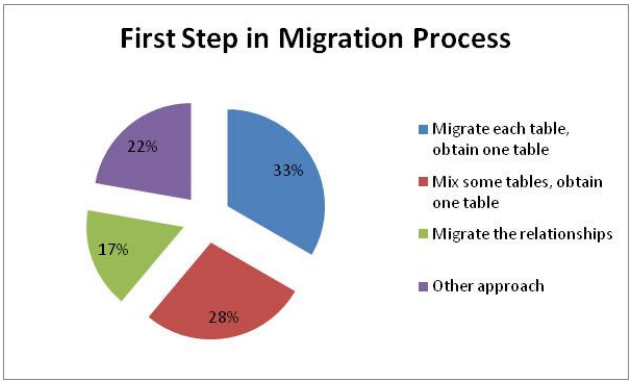


Figure 4. First step in the migration process.

The above argument is further reinforced by Fig. 6; it demonstrates that the majority of the participants (44%) had felt sometimes confused, i.e., without knowing how to go about it.

Fig. 7 provides the opinion of the participants in case that a set of guidelines it had been provided. 28% strongly agreed about their usefulness and 44% are agreeing with the relevance of this kind of tool in the migration process.

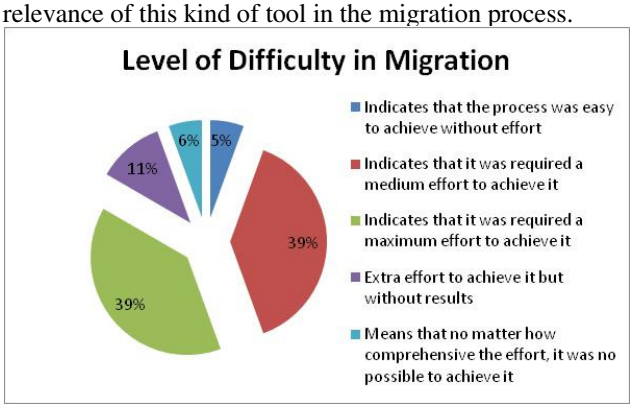


Figure 5. Level of difficulty in the migration process.

In the matters of the different database aspects, only five were studied in the experiment: tables, constraints, Primary Keys (PK), Foreign Keys (FK) and others (including all the aspects not specified in a clearly way such as fields, types of relationships, views, indexes, procedures and triggers).

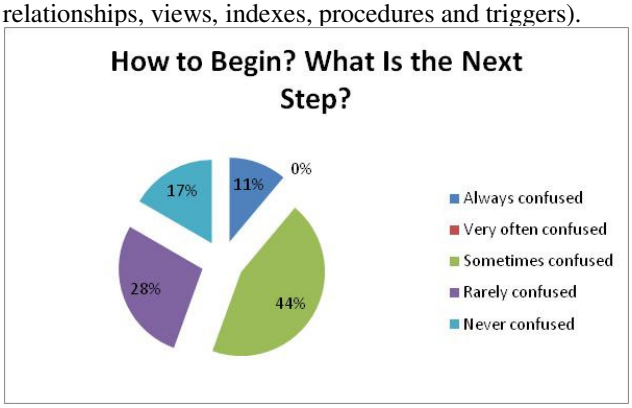


Figure 6. How to begin the process?.

Table IV reports the above information displaying the relational aspect covered against the percentage of coverage in terms of 0%, 25%, 50%, 75% and 100%. For instance, it is reveals that 50% of the participants think that their solution cover the relational aspect "tables" in a 100%. In contrast, 22% think that their solution covered this aspect in a 0%.

Moreover, Table IV presents that 28% of the participants think that their solution covers 100% the relational aspect "constraints". On the other hand, 39% of the participants think that their solution covered this aspect in a 0%.

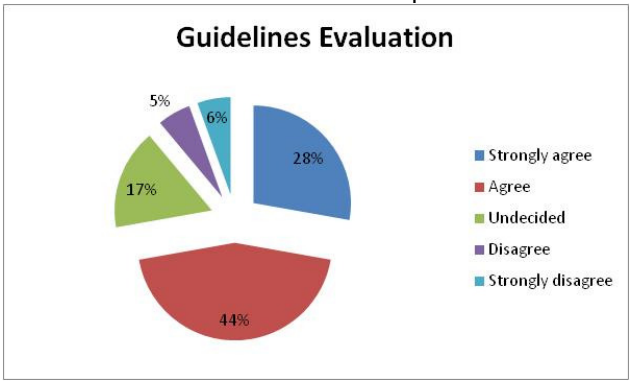


Figure 7. Guidelines evaluation.

TABLE IV. TABLE LEVEL OF COVERAGE IN DIFFERENT DB ASPECTS

Relational aspect covered	Percentage of coverage				
	0%	***25%***	***50%***	***75%***	***100%***
Table	22%	0%	6%	22%	50%
Constraint	39%	11%	17%	5%	28%
PK	29%	0%	18%	12%	41%
FK	28%	0%	11%	22%	39%
Others	94%	0%	0%	0%	6%

Furthermore, Table IV reports that 41% of the participants think that their solution covers 100% the relational aspect "primary keys". However, 29% think that their solution covered this aspect in a 0%.

Table IV shows that 39% of the participants think their solution covers 100% the relational aspect "foreign keys". Conversely, 28% think that their solution covered this aspect in a 0%.

Other relational database improvement aspect like fields, store procedures or triggers were put together in the relational aspect "others" and it reveals that 94% of the participants show no interest in these aspects. See Table IV, last row.

VII. CONCLUSION

The problem of migrate a relational schema from RDBMS to NoSQL environment was studied. We have developed a complete experiment that will allow a comparison with futures solutions approaches. This comparison will use the data obtained as a baseline. Besides, the data obtained in the experiment show the need for a formal and standard way to conduct this kind of migration. Despite the fact that today the migration is possible using heuristic techniques based on developer's experience, we have demonstrated that not all the staff with expertise in RDBMS has the enough expertise in NoSQL to carry out this kind of migration.

A surprising outcome of the experiment was that nearly all participants try to migrate the relational aspect "table" first, but they do not pay attention to other relational aspects like "relationships" or "fields". To explain this empirical finding, we investigated the background of the participants and we strongly believe, this result, was mainly caused by their large RDBMS experience.

Because this is the first step of the project, we cannot make a hypothesis testing, and much less, accept or reject the alternative hypothesis.

On the other hand, it is possible to show some feedback based on comments received during the workshop. Any information about guidelines was given to participants. It is reasonable to assume that those without familiarity in database have experienced more difficulties than others with some years of working with them.

The comments about the training session were positive in general. Despite the experiment trainer's effort, it can be observed that during the first half hour of the experiment there was a considerably spent of time consulting the reference documentation, especially those participants without the requested experience. According to the feedback of some PhD students, the first obstacle was to figure out what could be the first step to start the process. We expect to complete the all the research objectives in summer 2015.

VIII. FUTURE WORK

In near future, a more in-depth analysis will be presented. Besides, we will finish the guidelines set. The idea is refine the set with the advice of experts NoSQL users. Once the process of refinement is done, we will apply the same experiment with the same synthetic relational schema. At the end, the comparison explained in Section III will be conducted.

ACKNOWLEDGMENT

The authors gratefully acknowledge the ideas and support of David Lauzon and Khaled Almakadmeh from ÉTS software engineering department, and Jean-Marc Spaggiari, from Cloudera. Some of their ideas and perspectives were used in the experiment section. Special thanks also to Luis Bautista for his ideas about some solutions approaches. Besides, the authors want to express their gratitude for the guidance and invaluable technical assistance received from Patrice Dion, responsible for the technological platform of the software engineering laboratories in the ÉTS. Their help was invaluable to achieve this paper.

REFERENCES

[1] M. V. Zelkowitz, D. R. Wallace, and D. W. Binkley, "Experimental validation of new software technology", in Lecture notes on empirical software engineering. World Scientific Publishing Co., Inc. 2003. pp. 229-263.

[2] D. Abadi, “Data management in the cloud: limitations and opportunities. IEEE Data Engineering Bulletin, vol 32(1). pp 3-12. 2009.

[3] L. Chongxin, “Transforming relational database into HBase: a case study”. in Software Engineering and Service Sciences (ICSESS), IEEE International Conference on. 2010. pp 683-687.

[4] M. Stonebraker, S. Madden, D. Abadi, S. Harizopoulos, N. Hachem, and P. Helland, “The end of an architectural era: (it's time for a complete rewrite)”, in Proceedings of the 33rd international conference on very large data bases, VLDB Endowment: Vienna, Austria. 2007. pp. 1150-1160.

[5] M. Stonebraker, D. Abadi, D. DeWitt, S. Madden, E. Paulson, and A. Pavlo, “MapReduce and parallel DBMSs: friends or foes?” Commun. ACM, vol. 53(1). pp. 64-71. 2010.

[6] M. Stonebraker, “Technical perspective: One size fits all: an idea whose time has come and gone”. Commun. ACM. Vol. 51(12). pp. 76-76. 2008.

[7] J. Gantz and D. Reinsel, “Extracting Value from Chaos”, in EMC Corporation, E. Corporation, Editor. pp. 12-24. 2011.

[8] M. Stonebraker, “The case for shared nothing”. A quarterly bulletin of the IEEE computer society technical committee on database engineering, vol. 9(1). pp. 4-9. 1986.

[9] H. Boral and D. J. DeWitt, “A methodology for database system performance evaluation”. SIGMOD Rec.,vol. 14(2): pp. 176-185. 1984.

[10] J. Varia, “Migrating your existing applications to the AWS cloud”, Amazon Web Services. 2010.

[11] G. Demarest and R. Wang, “Oracle cloud computing”, Oracle, Editor, Oracle Redwood Shores, CA. 2010.

[12] H. Goto, Y. Hasegawa, and M. Tanaka, “Efficient scheduling focusing on the duality of MPL representatives,” Proc. IEEE Symp. Computational Intelligence in Scheduling (SCIS 07), IEEE Press, Dec. 2007, pp. 57-64, doi:10.1109/SCIS.2007.357670.

[13] D. Salmen, T. Malyuta, R. Fettersand, and A. Norbert, “Cloud data structure diagramming techniques and design patterns”. 2009.

[14] P. Singh, “Schema guidelines & case studies. 2010.

[15] T. Singh and P. S. Sandhu, “Cloud computing databases: latest trends and architectural concepts”. Proceedings of World Academy of Science, Engineering and Technology. vol. 73(Compendex). 2011. pp. 1042-1045.

[16] A. Jedlitschka and D. Pfahl. “Reporting guidelines for controlled experiments in software engineering”. in Proceedings of the 4th International Symposium on Empirical Software Engineering (ISESE 2005). IEEE Computer Society. 2005. Pp. 95-104.

[17] N. Juristo and A. M. Moreno, “Basics of software engineering experimentation”. Springer Publishing Company, Incorporated. 2010.

[18] C. Wohlin, M. Höst, and K. Henningsson, “Empirical research methods in software engineering”, in Empirical Methods and Studies in Software Engineering, R. Conradi and A. Wang, Editors. Springer Berlin Heidelberg. 2003. pp. 7-23.

[19] S. Easterbrook, J. Singer, M. A. Storey, and D. Damian, “Selecting empirical methods for software engineering research guide to advanced empirical software engineering”, in Guide to Advanced Empirical Software Engineering, F. Shull, J. Singer, and D. Sjøberg, Editors, Springer London. pp. 285-311. 2008.

[20] E. Marcos, “Software engineering research versus software development”. SIGSOFT Softw. Eng. Notes, vol. 30(4). 2005. pp. 1-7.

[21] M. Kasunic, “Designing an effective survey”. Carnegie Mellon Software Engineering Institut. 2005.

[22] T. C. Lethbridge, “A survey of the relevance of computer science and software engineering education”, in Proceedings of the 11th Conference on Software Engineering Education and Training. IEEE Computer Society. 1998. pp. 56-66.

 ISBN: 978-1-61208-388-9

Company Management Approaches — Stewardship or Agency: Which Promotes Better Security in Cloud Ecosystems?

Bob Duncan
Computing Science
University of Aberdeen
Email: bobduncan@abdn.ac.uk

Mark Whittington
Accounting and Finance
University of Aberdeen
Email: mark.whittington@abdn.ac.uk

***Abstract*—Historically, companies have been managed under the principles of agency theory. There is evidence to suggest that the complexity of modern computing systems, and in particular cloud computing systems, has become so convoluted that the principles of agency theory can no longer cope. We suggest that the adoption of stewardship theory for cloud security can present a possible credible alternative that can deliver much better results for the security of all cloud users, particularly in the long run.**

Keywords–Stewardship; agency; management; security; cloud computing.

I. INTRODUCTION

Modern information systems have evolved considerably over the past four decades, leading to the development of complex, highly distributed information systems and the need to police them properly. The need to address traditional security issues of confidentiality, integrity and availability (CIA) has increased this complexity further, due to the need for scalability and redundancy. As a consequence, despite the benefits offered by Moore's Law [1], costs have increased significantly. The statutory and regulatory environment is also ever increasing in reporting requirements, responsibilities and complexity, leading to an ever increasing additional cost burden. Fines for non-compliance are increasing year on year as regulators take a more and more aggressive approach. One regulator in the UK, the Financial Conduct Authority (FCA) [2] has so far levied fines of £1,427,943,800 during 2014. Contrast this with the level of fines levied by their predecessor, the Financial Standards Authority (FSA) [3], five years ago of £34,800,000.

Cloud computing offers the possibility of a substantial economic benefit to firms, yet at the same time, increases complexity and risk further. This results in an interesting dilemma. On the one hand, potential cost savings of 50 - 90% [4] are possible, which is highly attractive, but on the other hand, complexity can increase exponentially, placing significant increasing risk on business and government alike.

The practice of achieving compliance with recognised security standards is spreading. While this is a good idea in principle, in practice it may not provide the assurance being sought. The multiplicity of security standards currently available, and in particular, the lack of consensus on cloud security standards presents a difficult challenge. Compliance does not necessarily ensure protection. The procedures in use for achieving accreditation often vary enormously, with no real requirement for a rigorous approach. Checklists are often favoured over a deep and searching approach to security standards compliance [5], and there is generally no requirement for regular review.

Given the potential multiplicity of actors and the complexities of their relationships with each other in cloud ecosystems, it is clear that simple traditional agency relationships (where each actor looks to their own short term ends) will no longer be able to handle fully the security implications for users of these ecosystems. There is a clear need for developing a stronger mechanism to ensure users of such ecosystems can be assured of the security of their information.

The remainder of the paper is organized as follows: in Section II, we discuss three main barriers to effective cloud security; in Section III, we consider the merits of agency versus stewardship; in Section IV, we demonstrate how each impact on cloud security; and in Section V, we discuss our conclusions and future work.

II. THE CHALLENGES

We discuss three main challenges to be overcome when considering information security in the cloud - standards compliance, the limitations on management of agency theory and the sheer complexity of cloud ecosystems.

A. Standards

The recent development and rapid evolution of cloud ecosystems presents a far more rich and complex security environment than existing security mechanisms were designed to cope with. There is a danger that continued reliance on existing standards will lead to real weaknesses in systems which can be vulnerable to exploitation. The challenge here is to develop a means of identifying potential exposure in as comprehensive a manner as possible, yet be sufficiently flexible to adapt with a dynamically changing information ecosystem environment. There are a number of cloud security standards which have recently evolved, but the problem is, which standard should be used? Should it be the Association for Retail Technology Standards (ARTS) [6], the Cloud Security Alliance (CSA) [7], the Cloud Standards Organisation (CSCC) [8], the Distributed Management Task Force (DMTF) [9], the European Union Agency for Network and Information Security (ENISA) [10], the European Telecomminications Standards Institute (ETSI) [11], the Federal Risk and Authorization Management Program (FedRamp) [12], Generally Accepted Privacy Principles (GAPP) [13], the Global Inter-Cloud Technology Forum (GICTF) [14], the International Organization for Standardization (ISO) [15], the ITU Telecommunication Standardization Sector (ITU) [16], the National Insitute of Standards and Technology (NIST) [17], the Organization for the Advancement of Structured Information Standards (OASIS) [18], the Open Cloud Consortium (OCC) [19], the Open Grid

 ISBN: 978-1-61208-388-9

Forum (OGF) [20], the Object Management Gropu (OMG) [21], the Payment Card Industry (PCI) [22] or the Storage Networking Industry Association (SNIA) [23], to name but a few? None of these standards provides a comprehensive level of complete security — there is no "one size covers all", which also presents a limitation. Even compliance with every single standard will not guarantee complete security, which, in turn, presents yet another disadvantage [24]. While the universal implementation of such standards is often perceived as a laudable aim, there is a fundamental flaw where new computing technology is concerned, namely that the pace of evolution of new technology far outstrips the capability of international standards organisations to keep up with the changes [25]. In addition to which, latest estimates [26] suggest that over 200,000 new malware threats are being developed globally every day. We addressed this in earlier work [38][24].

B. Agency

Another issue arises where the principals of companies utilise the principles of corporate governance [27][28], based on agency theory. Jensen and Meckling [29] recognized that while both principal and agent were utility maximizers, they would not necessarily always have the same alignment of goals. Further, the agent is more likely to have complete knowledge, whereas the principal's generally is incomplete, and this can disadvantage the principal, or at least require the expenditure of additional sums to try to safeguard the position of the principal. Over time, agents, having complete information, can make more decisions which do not fully benefit the principal, resulting in better utility for themselves. It is very rare that the goals of principal and agent will perfectly align, thus gaining mutual satisfaction, and this is the fundamental flaw agency theory highlights. The business environment is constantly changing, as are corporate governance rules, with more emphasis now being placed on responsibility and accountability [30], social conscience [31], sustainability [32][33], resilience [34] and ethics [35]. We focus on this challenge in this paper.

C. Complexity

Yet another issue is the increasing complexity which new technology brings, and the ever increasing potential exposure to risk brought about by a failure to grasp the significance of risks arising as a result of this increase in complexity [36]. Traditional distributed information systems present a multiplicity of technical layers, each of which must interact with one or more other layers. Rather than simplifying this process, cloud introduces yet more layers. There is Infrastructure, Platform and Software as a Service (IaaS, Paas and SaaS), each of which can be operated by different actors. Cloud brokers may also be involved, leading to yet more layers, yet more complexity, yet more risk. Thus, there is a need for a more agile, effective, approach to address these issues. Another hurdle to be overcome is the cross disciplinary nature of today's corporate world. There is more cross-over between disciplines than in the past, which means no single discipline can effectively deal with all the issues arising from the use of cloud technology [37]. Existing security paradigms have not kept pace with the rapidity of development, change and complexity in modern information ecosystems. There is a danger that continued reliance on existing models will lead to real weaknesses in systems which can be vulnerable to exploitation. The challenge here is to develop a means of addressing these weaknesses at a conceptual level which can be demonstrably more robust than existing mechanisms currently in place. We address this challenge in our future work.

III. Agency vs Stewardship

In order to compare the relative merits of agency against stewardship, it is necessary to understand more clearly what each is and how they work. In the following sub sections, we outline a definition of each and their limitations, followed by a comparison.

A. The Agency Theory of Management

Management has been a focus of academic study for a great many years. Initially, companies were managed by their owners, which is still the model today for many fledgling small businesses. Over time, companies grew larger and larger, resulting in their becoming too large for an individual to be able to provide them with sufficient capital to meet their needs and cover their potential liabilities. The root of this problem can be traced back to the modern corporation, as discussed by Berle and Means [39], creating a separation between ownership and control of wealth. While owners would generally prefer to manage and control their own companies to maximize their own utility for themselves, the large scale of the "modern corporation" puts information management, massive capital needs and economic obligations far beyond the reach of the individual. This increase in size, and capital requirements, led to companies being managed by professional managers (agents) on behalf of the company owners (principals) which led to the development of agency theory.

B. The Definition of Agency Theory

Jensen and Meckling [29] provide us with a definition of agency theory: *"We define an agency relationship as a contract under which one or more persons (the principal(s)) engage another person (the agent) to perform some service on their behalf which involves delegating some decision making authority to the agent. If both parties to the relationship are utility maximizers, there is good reason to believe that the agent will not always act in the best interests of the principal. The principal can limit divergences from his interest by establishing appropriate incentives for the agent and by incurring monitoring costs designed to limit the aberrant activities of the agent. In addition in some situations it will pay the agent to expend resources (bonding costs) to guarantee that he will not take certain actions which would harm the principal or to ensure that the principal will be compensated if he does take such actions".*

Some examples of this relationship are:

- The electorate (principal) and government (agent);
- Shareholders (principal) and Chief Executive Officer (agent);
- Employers (principal) and employees (agent);
- Contractee (principal) and contractor (agent).

There is not a single relationship within a company. Rather the relationship cascades through the organisation and into the outside world. Thus, for example, shareholder to chief executive officer (CEO), CEO to business managers, business managers to staff, company to suppliers, customers to company, cloud user to cloud service provider and every sub contracted relationship.

 ISBN: 978-1-61208-388-9

C. The Limitations of Agency Theory

Jensen and Meckling recognised that while both principal and agent were utility maximisers, they would not necessarily always have the same alignment of goals. Further, the agent is more likely to have complete knowledge, whereas the principal generally has incomplete knowledge, and this can disadvantage the principal, or at least require the expenditure of additional sums to try to safeguard the position of the principals. Over time, agents, having complete information, can make more and more decisions which do not fully benefit the principals, resulting in better utility for themselves. It is very rare that the goals of principal and agent will perfectly align, resulting in mutual satisfaction, and this is both the fundamental insight and flaw in relying upon agency for successful delegation.

D. The Stewardship Theory of Management

The implications of agency theory, agents adhering to the terms of their contract without necessarily achieving the principal's desired outcomes, are problematic and the literature has considered the more principle-based stewardship approach. This has been discussed over several decades, across a number of disciplines, such as accounting [40][41], management research [42][43][44], information stewardship [45][46], where Pym et al specifically focus on cloud stewardship, and in natural resource management [34], where Chapin et al demonstrate, using a systems view, the benefits of the stewardship approach, as does Kao [47].

E. The Definition of Stewardship Theory

Davis [43] provides a good definition of stewardship theory:

"In stewardship theory, the model of man is based on a steward whose behavior is ordered such that pro-organizational, collectivistic behaviors have higher utility than individualistic self-serving behaviors. Given a choice between self-serving behavior and pro-organizational behavior, a steward's behavior will not depart from the interests of his or her organization. A steward will not substitue or trade self-serving behaviors for cooperative behaviors. Thus, even where the interests of the steward and the principal are not aligned, the steward places higher value on cooperation than defection (terms found in game theory). Because the steward perceives greater utility in cooperative behavior and behaves accordingly, his or her behavior can be considered rational.

According to stewardship theory, the behavior of a steward is collective, because the steward seeks to attain the objectives of the organization (e.g., sales growth or profitability). This behavior in turn will benefit principals such as outside owners (through positive effects on profits on dividends and share prices) and also pricipals who are managerial superordinates, because their objectives are furthered by the steward. Stewardship theorists assume a strong relationship between the success of the organization and the principal's satisfaction. A steward protects and maximises shareholders' wealth through firm performance, because, by so doing, the steward's utility functions are maximised".

F. The Limitations of Stewardship Theory

The limitations found in agency theory do not apply in stewardship theory. Since the utility of the steward is firmly in alignment with the utility of the principals, this removes the temptation to make decisions solely for the benefit of the steward. Any decision that benefits the principals will also benefit the steward, and conversely any decision that benefits the steward will also benefit the principals. The only proviso here is the need to provide a sufficient means to incentivise the business manager to become a steward, rather than a self-serving agent.

G. Stewardship Synergy with Cloud Ecosystems

There is a natural synergy between stewardship and cloud ecosystems [48]. Cloud ecosystems are dependent on the building and maintaining of robust relationships between all the actors in the ecosystems [49]. This dependency arises out of a need for sustainability, resilience and ethicality. In order for a greater take up of cloud usage, there needs to be trust and a mutual accountability between all the actors involved. The multiplicity of actor relationships and this need for responsibility and accountability means that the traditional agency approach cannot succeed. The cloud ecosystem is too rich an environment for the agency approach to be able to succeed, whereas stewardship is tailor made to handle this level of complexity [50].

The EU recognizes the existence of this complexity in relationships, especially with regard to information security in the cloud, and has produced a working paper [51] for discussion on the subject. The ISO 27000 standards, while they address the notion of security, are not yet sufficiently well developed to fully cover these issues. There is no doubt they will be expanded to cover these issues in time. They do recognise the existence of corporate outsourcing, but as yet this has not been fully adapted to cover all the modern extensions of this mechanism, including off-shoring and cloud.

We believe a stewardship approach represents an ideal mechanism to address the shortcomings which presently exist. This approach may provide a useful means to help businesses to adopt cloud more readily, to better reap the benefits and economies offered, while maintaining a better grasp of the security implications associated with such a move.

H. Why the Time is Right for Change

The culmination of years of self-serving behaviour on the part of managers has led to more extreme agent behaviour [52]. Also, it leads to a short term view of running a business, and this can work against the long term sustainability of the business and impact adversely on resilience. It can also lead to driving managers into behaving less ethically due to these pressures to perform in the short term. Equally, the agency behaviour of large scale shareholders has helped to encourage this behaviour in managers, as these shareholders are frequently looking for the best short term returns. Thus, the effects of greed by both managers and certain shareholders seem to take agency theory to a logical extreme. There is no mechanism in agency theory to deal with the broad themes of sustainability, resilience and ethicality. It is no coincidence that this behaviour is particularly prevalent in the banking industry.

During the past 15 years, Enron and other scandals led to the passing of the Sarbanes-Oxley Act [53] in the United States of America (USA). In 2008, the banking crisis arrived, with all the attendant fall out. There have been countless corporate frauds of some magnitude, such as the Madoff scandal. There is a perception among shareholders that the prescriptions to

 ISBN: 978-1-61208-388-9

deal with agency theory no longer work to reign in the worst excesses of corporate management [52].

Indeed, in the five years prior to the financial crisis of 2008, the annual report of the Royal Bank of Scotland (RBS) made much of their stewardship of the business. In the 2007 annual report [54] published in spring 2008, the then Sir Fred Goodwin used stewardship language in his Group Chief Executive's review, stating "Our results demonstrate the resilience of the Group in the face of testing circumstances." These claims of stewardship were clearly a sham, as that resilience rather catastrophically ran out on 7 October 2008.

Corporate managers themselves are beginning to see the effect their "fat cat" bonuses and incentive schemes are having on shareholders, to the point that many are now voluntarily agreeing to reduce or even give up their entitlement, especially where the business does not perform so well. Shareholders are looking for a return to "the good old days", when corporate managers were looked on as honourable people, who put the interests of the business first, and their own interests second. In essence, they embodied many of the core values of stewardship.

I. Agency vs Stewardship Summary

In Table I, we can see the major differences between agency and stewardship, the psychological and situational mechanisms involved.

Since the utility of the steward is in alignment with the utility of the principal, this removes the temptation to make decisions solely for the benefit of the steward. Any decision that benefits the principal will also benefit the steward. The stakeholders and relationships in a business are not, of course, limited to managers and shareholders. Customers, suppliers, government, audit firms and even the local communities are stakeholders in the business. As noted above, in corporate governance today, we see much more consideration being given to the the notion of corporate social responsibility, resilience, sustainability and an ethical approach to doing business. There is certainly more pressure on managers in today's business world to take a more outward view of their actions, potentially leading to a more responsible stewardship approach. There is an ever growing appetite for more accountability in business, being driven by shareholders, government, customers, suppliers, auditors and the general public alike.

TABLE I. A COMPARISON OF AGENCY AND STEWARDSHIP[43]

	Agency Theory	**Stewardship Theory**
Model of man	Economic man	Self-actualizing man
Behaviour	Self-serving	Collective serving
Psychological Mechanism		
Motivation	Lower order/economic needs (psychological, security, economic)	Higher order needs (growth, achievement, self-actualization)
Social Comparison	Extrinsic Other managers	Intrinsic Principal
Identification	Low value committment	High value committment
Power	Institutional (legitimate, coercive, reward)	Personal (expert, referent)
Situational Mechanism		
Management Philosophy	Control oriented	
Risk orientation	Control oriented	Involvement oriented
Time frame	Short term	Long term
Objective	Cost control	Performance enhancement
Cultural Differences	Individualism High power distance	Collectivism Low power distance

It is clear that the culmination of years of self-serving behaviour on the part of business managers has led to agency behaviour moving beyond its own limitations. It would seem the fundamental flaw with agency theory is that it cannot cope with unbridled greed. Also, it leads to a short term view of running a business, and this can work against the long term sustainability of the business and impact adversely on resilience. It can also lead to driving managers into behaving less ethically due to these pressures to perform in the short term. Equally, the agency behaviour of large scale shareholders has helped to encourage this behaviour in managers as the shareholders are frequently looking for the best short term returns.

Thus, the effects of greed on both the part of managers and shareholders push agency theory beyond its capabilities. There is no mechanism in agency theory to deal with sustainability, resilience and ethicality. At the time agency theory was developed, such concerns were of little relevance, but in today's business world, that is no longer the case.

In Table II, we can see how the major differences between adopting agency and stewardship affect the relationship between the manager and the shareholders.

TABLE II. A PRINCIPAL-MANAGER CHOICE MODEL[43]

Principal's Choice

Manager's Choice	*Agent*	*Steward*
Agent	Minimize Potential Costs Mutual Agency Relationship 1	Agent Acts Opportunistically Principal is Angry Principal is Betrayed 2
Steward	3 Principal Acts Opportunistically Manager is Frustrated Manager is Betrayed	4 Maximize Potential Performance Mutual Stewardship Relationship

Where manager and shareholder both adopt the agency approach, there can be mutual satisfaction as long as their respective goals are in alignment, but, as has already happened in many cases, over time this can lead to a mutually destructive relationship developing.

Where manager and shareholder adopt an opposite approach, with one adopting an agency approach and the other adopting a stewardship approach, there will be a disparity between the outcomes for each, regardless of which adopts which position.

Where both manager and shareholder alike are prepared to adopt a stewardship approach, their joint goals are much better aligned, and the extreme pressures and destructive nature of the short-term approach can be set aside. This mutually beneficial approach also serves to handle the additional requirements of sustainability, resilience and ethicality for the business, which will benefit both parties. This allows for a long-term view to be developed by all concerned, which will result in a far better outcome for everyone.

 ISBN: 978-1-61208-388-9

IV. IMPACT ON CLOUD SECURITY

So, the question is how does management approach impact on cloud security?

Cloud service providers (CSP) have developed their cloud business models using agency theory. Standard service level agreement (SLA) offerings from the major players basically ignore accountability, assurance, audit, compliance, integrity, privacy and security, merely offering availability as the focus of their measure of performance. The onus for measuring and proving unacceptable performance is neatly passed to the customer, which, with the inclusion of some suitably deeply buried clauses in the small print, assures the buck invariably never stops with the CSP.

Of course it is possible to negotiate an SLA to include these missing measures, but you can be sure the cost of good corporate lawyers and the increased service costs involved will decimate any potential cost savings offered by the cloud paradigm. Since such costs would, in any event, only be affordable by the largest corporations, this puts most small and medium-sized enterprises (SME) and sole traders at a commercial disadvantage.

This is clearly worrying in today's climate of increasing punitive regulatory fines for privacy and security breaches and the potential negative impact on business costs and the knock-on negative impact on share values. Taken against a backdrop of an ever expanding threat environment, it is clear that positive action is needed globally.

Compliance with security standards can be viewed as an agency reaction by company management to protect them from being sued by their own principals for failing to implement proper security. Since current standards are neither complete, nor up to date, compliance with these standards cannot ensure security [24].

In Table III, we can see how the Cloud User - CSP Choice Model can lead to an improvement in this situation. CSPs can no longer afford to "stick their heads in the cloud" when it comes to information security. There is no "I" in team. Every actor in the cloud ecosystem needs to contribute towards better security for society as a whole. Growth in global security breaches increased by more than twice the rate of global gross domestic product (GDP) increase during 2014 [55]. Failure to tackle this issue means this situation will only get worse.

TABLE III. CLOUD USER-CSP CHOICE MODEL[43]

		Cloud User(Principal)'s Choice	
		Cloud User(Agent)	*Cloud User(Steward)*
	CSP(Agent)	Minimize Potential Costs Mutual Agency Relationship Security is fair 1	CSP Acts Opportunistically Cloud User is Angry Cloud User is Betrayed Security is poor 2
CSP's Choice	*CSP(Steward)*	3 Cloud User Acts Opportunistically CSP is Frustrated CSP is Betrayed Security is poor	4 Maximize Potential Performance Mutual Stewardship Relationship Security is good

Every actor in the cloud ecosystem is responsible for maintaining good security. That means in addition to top management, middle and lower management must equally be committed, as must all the company's employees. Company suppliers and company customers too must play their part. A major component of the cloud ecosystem is of course the service providers who provide every element of the system, whether they are providing IaaS, PaaS, SaaS or some other aspect "as a service". All actors must be prepared to be accountable for ensuring a proper level of security is maintained and every relationship has the potential to be managed as an agency or stewardship one. Service providers must be prepared to ensure SLAs can be re-written to more fairly spread risk and accountability between all the actors to ensure better security of the whole. Company auditors must also recognise the risk involved in the use of cloud systems and must be more vigilant to ensure areas of weakness are highlighted and dealt with properly in good time. The threat environment is not going to improve for the better any time soon.

V. CONCLUSION AND FUTURE WORK

Thus, we can see that every member of the cloud ecosystem needs to be more aware of their role within the system. Each has their part to play. There are a great many potential weaknesses in cloud ecosystems. While these can be identified and addressed to ensure proper levels of security can be achieved and maintained, there is no doubt that as long as the agency approach persists, with all the actors pursuing their own agendas, it will be extremely difficult to achieve a satisfactory level of security.

It is certainly the case that the extremes of agency behaviour can lead to lower levels of security being achieved within cloud ecosystems. Conversely, a stewardship approach can lead to a more robust security stance, which can provide additional resilience and sustainability in the long run for a company. Given the rapidly evolving nature of the threat environment, it is no longer a question of if a company will be compromised, but rather it is a question of when, and for how much? There is also the question of how much damage a compromise will do to the reputation of the company, and the possible impact on the share price.

We would argue that a shift from agency behaviour to a stewardship approach can go a long way to reducing the major weaknesses inherent in an agency approach to security in cloud ecosystems. We would also argue that following decades of corporate excess brought about by the worst of agency behaviour, it would be appropriate for corporate management to consider taking a step change in their outlook towards a stewardship based management style. This would not only promote long term sustainability and resilience of companies, but would lead to a more honest approach to information security. We would further argue that this will require a significant change in attitude from the cloud service providers, leading to the development of better security oriented SLAs, which will improve the approach to security for all actors within the cloud ecosystem.

The next step is to consider how a stewardship approach, requiring a high level of trust, can be encouraged as seeming practical rather than naïve; this will inevitably require increased transparency and the welcoming of high quality monitoring by all sides.

 ISBN: 978-1-61208-388-9

REFERENCES

[1] G. E. Moore, "Cramming More Components onto Integrated Circuits," Electronics, no. April 19, 1965, pp. 114-117.

[2] FCA, "Fines Table - 2014," 2014. [URL]: http://www.fca.org.uk/firms/being-regulated/enforcement/fines retrieved: Jan 2015

[3] P. Taylor, "FSA fines double to record 35m in 2009," 2009. [URL]: http://www.telegraph.co.uk/finance/newsbysector/banksandfinance/6852385/FSA-fines-double-to-record-35m-in-2009.html retrieved: Jan 2015

[4] P. Mell and T. Grance, "Effectively and Securely Using the Cloud Computing Paradigm," Tech. Rep., 2009.

[5] B. Duncan and M. Whittington, "Reflecting on Whether Checklists Can Tick the Box for Cloud Security." in Cloud Comp. Tech. Sci. (CloudCom), 2014 IEEE 6th Int. Conf. Singapore: IEEE, 2014, pp. 1-6.

[6] ARTS, "Association for Retail Technology Standards," 2014. [URL]: https://nrf.com/resources/retail-technology-standards-0 retrieved: Jan 2015

[7] CSA, "Security Guidance for Critical Areas of Focus in Cloud," Cloud Security Alliance, Tech. Rep., 2012.

[8] CSO, "Cloud Standards," 2013. [URL]: http://cloud-standards.org/ retrieved: Jan 2015

[9] DMTF, "Distributed Management Task Force: Standards and Technology," 2014. [URL]: http://www.dmtf.org/standards retrieved: Jan 2015

[10] ENISA, "A Security Analysis of Next Generation Web Standards," 2013. [URL]: http://www.enisa.europa.eu/ retrieved: Jan 2015

[11] ETSI, "European Telecommunications Standards Institute," 2014. [URL]: http://www.etsi.org/standards retrieved: Jan 2015

[12] FedRamp, "FedRamp," 2014. [URL]: http://cloud.cio.gov/fedramp retrieved: Jan 2015

[13] CPA/CA, "Generally Accepted Privacy Principles," Tech. Rep., 2009.

[14] GICTF, "Global Inter-Cloud Technology Forum," 2012. [URL]: http://www.gictf.jp/index_e.html retrieved: Jan 2015

[15] ISO.org, "ISO/IEC 27000:2009 - Information technology - Security techniques - Information security management systems - Overview and vocabulary," ISO.org, Geneva, Switzerland, Tech. Rep., 2009.

[16] ITU, "ITU Telecommunication Standardization Sector," 2014. [URL]: http://www.itu.int/en/ITU-T/publications/Pages/default.aspx retrieved: Jan 2015

[17] NIST, "NIST Security and Privacy Controls for Federal Information Systems and Organizations," Natioinal Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. February, 2012.

[18] OASIS, "Organization for the Advancement of Structured Information Standards," 2014. [URL]: https://www.oasis-open.org/standards retrieved: Jan 2015

[19] OCC, "Open Cloud Consortium," 2014. [URL]: http://opencloudconsortium.org/working-groups/ retrieved: Jan 2015

[20] OGF, "Open Grid Forum," Tech. Rep.

[21] OMG-CC, "Object Management Group," 2014. [URL]: http://www.cloud-council.org/ retrieved: Jan 2015

[22] P. S. S. Council, "Data Security Standard Requirements and Security Assessment Procedures," PCI Security Standards Council, Tech. Rep. November, 2013.

[23] SNIA, "Cloud Data Management Interface," Tech. Rep., 2010.

[24] B. Duncan and M. Whittington, "Compliance with Standards, Assurance and Audit: Does this Equal Security?" in Proc. 7th Int. Conf. Secur. Inf. Networks. Glasgow: ACM, 2014, pp. 77-84.

[25] G. T. Willingmyre, "Standards at the Crossroads," StandardView, vol. 5, no. 4, 1997, pp. 190–194.

[26] Kaspersky, "Global Corporate IT Security Risks : 2013," Tech. Rep. May, 2013.

[27] S. Ross, "The Economic Theory of Agency: The Principal's Problem," Am. Econ. Rev., vol. 63, no. 2, 1973, pp. 134-139.

[28] M. Eisenhardt, "Agency Theory : An Assessment and Review," Acad. Manag. Rev., vol. 14, no. 1, 1989, pp. 57-74.

[29] M. C. Jensen and W. H. Meckling, "Theory of the Firm: Managerial Behavior, Agency Costs and Ownership Structure," J. financ. econ., vol. 3, no. 4, 1976, pp. 305-360.

[30] M. Huse, "Accountability and Creating Accountability: a Framework for Exploring Behavioural Perspectives of Corporate Governance," Br. J. Manag., vol. 16, no. s1, Mar. 2005, pp. S65-S79.

[31] A. Gill, "Corporate Governance as Social Responsibility: A Research Agenda," Berkeley J. Int'l L., vol. 26, no. 2, 2008, p. 452.

[32] C. Ioannidis, D. Pym, and J. Williams, "Sustainability in information stewardship: Time Preferences, Externalities and Social Co-Ordination," in WEIS 2013, 2013, pp. 1-24.

[33] A. Kolk, "Sustainability, Accountability and Corporate Governance: Exploring Multinationals' Reporting Practices," Bus. Strateg. Environ., vol. 17, no. 1, 2008, pp. 1-15.

[34] I. F. Stuart Chapin, G. P. Kofinas, and C. Folke, Principles of Ecosystem Stewardship: Resilience-Based Natural Resource Management in a Changing World. Springer, 2009.

[35] S. Arjoon, "Corporate Governance: An Ethical Perspective," J. Bus. Ethics, vol. 61, no. 4, Nov. 2005, pp. 343-352.

[36] E. Zio, "Reliability engineering: Old problems and new challenges," Reliab. Eng. Syst. Saf., vol. 94, no. 2, Feb. 2009, pp. 125-141.

[37] M. Dlamini, J. Eloff, and M. Eloff, "Information security: The moving target," Comput. Secur., vol. 28, no. 3-4, May 2009, pp. 189-198.

[38] B. Duncan, D. J. Pym, and M. Whittington, "Developing a Conceptual Framework for Cloud Security Assurance," in Cloud Comp. Tech. Sci. (CloudCom), 2013 IEEE 5th Int. Conf. Bristol: IEEE, 2013, pp. 120-125.

[39] A. A. Berle and G. C. Means, The Modern Corporation and Private Property, 1932.

[40] F. y. Gjesdal, "Accounting for Stewardship," J. Account. Res., vol. 19, no. 1, 1981, pp. 208-231.

[41] V. O'Connell, "Reflections on Stewardship Reporting," Account. Horizons, vol. 21, no. 2, Jun. 2007, pp. 215-227.

[42] L. Donaldson, "Stewardship Theory or Agency Theory: CEO Governance and Shareholder Returns," Aust. J. Manag., vol. 16, no. June 1991, 1991, pp. 49-65.

[43] J. H. Davis, F. D. Schoorman, and L. Donaldson, "Toward a Stewardship Theory of Management," Acd. Mgt. Rev., vol. 22, no. 1, 1997, pp. 20-47.

[44] C. E. Crutchley and R. S. Hansen, "A Test of the Agency Theory of Managerial Ownership, Corporate Leverage, and Corporate Dividends," Financ. Manag., vol. 18, no. 4, Jan. 1989, pp. 36-46.

[45] P. S. Licker, "Application Stewardship: A User Responsibility Approach to Post-Implementation Application Performance," MIS Q., 2010, pp. 151-157.

[46] D. Pym, M. Sadler, S. Shiu, and M. C. Mont, "Information Stewardship in the Cloud : A Model-Based Approach," Proc. CloudComp, 2011, pp. 1-20.

[47] R. Kao, Stewardship Based Economics. World Scientific, 2007.

[48] A. Baldwin, D. Pym, M. Sadler, and S. Shiu, "Information Stewardship in Cloud Ecosystems: Towards Models, Economics, and Delivery," 2011 IEEE Third Int. Conf. Cloud Comput. Technol. Sci., Nov. 2011, pp. 784–791.

[49] D. Pym, M. Sadler, S. Shiu, and M. C. Mont, "Information Stewardship in Cloud Computing," Int. J. Serv. Sci. Manag. Eng. Technol., vol. 1, no. 1, 2010, pp. 50-67.

[50] C. Ioannidis, D. Pym, and J. Williams, "Fixed Costs , Investment Rigidities , and Risk Aversion in Information Security : A Utility-theoretic Approach," 2013, pp. 1-16.

[51] EU, "Unleashing the Potential of Cloud Computing in Europe," 2012. [URL]: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=SWD:2012:0271:FIN:EN:PDF retrieved: Jan 2015

[52] J. Harris, "Whats Wrong with Executive Compensation? Jared," J. Bus. Ethics, vol. 85, no. 1, 2009, pp. 1-22.

[53] SOX, "Sarbanes-Oxley Act of 2002," p. 66, 2002. [URL]: news.findlaw.com/hdocs/docs/gwbush/sarbanesoxley072302.pdf retrieved: Jan 2015

[54] RBS, "Royal Bank of Scotland Group Annual Report Year Ending 2007," Royal Bank of Scotland, London, Tech. Rep., 2007.

[55] PwC, "The Global State of Information Security," PwC, Tech. Rep., Jan. 2015. [URL]: http://www.pwc.com/gx/en/consulting-services/information-security-survey/download.jhtml retrieved: Jan 2015

 ISBN: 978-1-61208-388-9

A Benchmarking Based SLA Feasibility Study Method for Platform as a Service

Ge Li, Frédéric Pourraz, Patrice Moreaux

Université Savoie Mont Blanc, Annecy le vieux, France
Email: {ge.li, frederic.pourraz, patrice.moreaux}@univ-savoie.fr

***Abstract*—Service Level Agreements (SLA) are contracts established between Software as a Service (SaaS) providers and Platform as a Service (PaaS) providers related to various properties of the services running on the platform. We propose a generic method to evaluate to what extent SaaS provider proposed Quality of Service (QoS) targets, such as response time or maximal throughput, can be guaranteed with constraints on workload, resources and cost. These "what-if" evaluations are based on small scale benchmarking and ability of the application to be scaled. The approach allows us to estimate with minimal costs, to what extend a given SLA can be accepted by both client and provider of a given PaaS.**

Keywords–SLA; Capacity planning; SLA negotiation; Benchmark; PaaS.

I. INTRODUCTION

Cloud services [1] can be classified as Infrastructure as a Service (IaaS), PaaS and SaaS. SaaS providers rely on PaaS features to benefit from the "pay as you go" paradigm corresponding to pay only "adapted" resources usage at runtime while satisfying a set of constraints related to QoS, workload and cost. Elasticity [2] is the property of the platform to meet all these requirements at a given variation rate. SLAs are used to describe the responsibilities of contracting parties (SaaS provider and PaaS provider). SLA management involves two stages [3]: establishing an SLA before runtime, and fulfilling the SLA at runtime. Establishing SLA is usually done through an automated negotiation process. It involves benchmarking and/or modelling the system (application and runtime support) to define the levels of QoS accepted by the SaaS "client" and offered by the PaaS provider and the constraints both parties will fulfil.

Web Service Level Agreement (WSLA), WS-Agreement and other SLA models proposed in SLA@SOI are significant proposals about SLA [4]. However, more work has to be done to take into account SLA in the context of cloud computing [3], especially at runtime. PSLA [5] is a PaaS level SLA description language based on WS-Agreement, which is an extendible SLA skeleton. PSLA takes the particular needs in PaaS level SLA into consideration and is well structured, commonly usable and machine readable for PaaS level SLA management. Modelling arrival rate and resource demands when satisfying QoS targets are foundations of capacity evaluation. Modelling methods can be classified as simulation modelling and analytic modelling.

As it is the case for most of current applications deployed in cloud infrastructures, we assume that the architecture of the application (App) is given as a graph of components (whatever the component model used) and that the designer of App has mapped this architecture on a graph of stages. Vertices of the graph are the components and edges are the relationships service required - service provided between components. A stage s is then instantiated as identical VMs at runtime. Horizontal scaling (HS), or scaling out, of s, means that s can be instantiated by several VMs (with a front-end Load Balancer (LB)). Vertical scaling (VS), or scaling up, of s means that VMs with different resources (for instance 1 to 3 CPUs, 1 to 4 GB of RAM), termed flavors, can instantiate s. Hence, a configuration of App is the deployed set of VMs of its graph of stages according to the HS and/or VS capability of the stages. Since we want to design a non intrusive method with respect to the application, our model will be based on results of benchmarks run in the target PaaS context, using measures also available at runtime.

Our method is as follows. First, we check to what extent the proposed SLA can be meet. To do so, we benchmark various configurations of the application. If the application and the underlying IaaSs on which the configuration is deployed can achieve the SLA, then we control the running application based on a set of configurations derived from the first step and on monitoring both the submitted workload (user's requests) and the behaviour of the running application. In this paper, we address the first step of the method and we propose an application independent, cost effective, benchmarking based, SLA feasibility study method from the PaaS provider's perspective. Since the number of feasible configurations may be "large" (more than 100) and time for benchmarking can be from minutes to hours, we propose to reduce the number of benchmarks by selecting the set of configurations benchmarked.

The paper is organized as follows. Section II briefly reminds the reader benchmarking methodology. Then Section III presents the context of our approach. In Section IV, we describe in detail our benchmark based SLA feasibility study method. Finally, we conclude this paper in Section V.

II. RELATED WORK

Benchmarking methods [6] are a well known set of methods to analyse the behaviour of software and hardware. The increasing number of software systems running in cloud infrastructures has raised a new interest in benchmarking because of the specificities of the running environments of these systems and of the variety of the applications. In [7], Iosup et al., summarise recent work on benchmarking cloud applications and propose to adapt benchmarking methods to cope with properties of these applications, especially for what concerns elasticity and variability. Due to the fact that controlling the runtime cloud application can lead to modify its allocated resources and since the application uses resources which are usually shared with other applications under control of hypervisors, benchmarking such applications must address the behaviour of the application faced up to these endogenous and exogenous variations [8].

III. SLA FEASIBILITY STUDY CONTEXT

SLA feasibility study is based on the evaluations of elasticity constraints described in PSLA. QoS target quantifies

 ISBN: 978-1-61208-388-9

TABLE I. SLA FEASABILITY STUDY

(a) FLAVOR DESCRIPTION

FLAVOR	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram
vCPU	8	5	6	4	5	3	1
RAM(G)	2.67	1.67	3	2	10	6	2
DISK(G)	13.35	8.35	15	10	25	15	5
COST(euro/s)	5.34	3.34	3	2	5	3	1

(b) RESOURCE CONSTRAINTS: MAXIMUM NoI

MAX_NoI	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram
Apache			2	3			
Jonas	4	6					
Mysql	2	3	2	3	2	3	4

(c) RESOURCE CONSTRAINTS: MINIMUM NoI

MIN_NoI	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram
Apache			1	1			
Jonas	2	2					
Mysql	1	1	1	1	1	1	1

(d) MFC(TIER,1,FLAVOR).

MFC(*,1,*)	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram
Apache			300	200			
Jonas	150	94					
Mysql	134	84	150	100	400	300	100

(e) MFC(TIER,MAX_NoI,FLAVOR).

MFC(*,MAX_NoI,*)	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram	MAX_MFC(TIER)
Apache			600	600	400			600
Jonas	600	564						600
Mysql	268	252	300	300	800	900	400	900

(f) MINIMUM COST TO SERVE MAR

COST	L_cpu	M_cpu	L_gnrl	M_gnrl	L_ram	M_ram	S_ram
Apache			3	4			
Jonas	10.68	13.36					
Mysql	16.02	13.36	6	6	5	3	3

the acceptable QoS metric "Value Range", e.g., Response time $\leq$ 13s. Workload constraints include "Data Size", "Data Composition"; e.g., it is composed of only "atomic request A" with a Maximum Arrival Rate (MAR) of 300 requests/s. Resource constraints indicate the kinds of allowed scaling actions and its limits; e.g., we indicate the minimum (in Table I(c)) and maximum (in Table I(b)) Number of Instances (NoI) for flavors (a flavor is a set of resources allocated to a virtual machine) (in Table I(a)) and tiers, which can be used for scaling actions. Cost constraints tell how much money can be spent per time unit, e.g., cost $<$ 10 euro/s.

Scaling actions for virtualized application are at the virtual machine level. To benchmark the application, we use "CLIF is a Load Injection Framework" (CLIF) and SelfBench services, which are provided as services under OpenCloudware project. CLIF [9] is designed for generating predefined traffic on a system to measure QoS target and observe the computing resources usage at the same time. CLIF can be used for deployment, remote control, monitored measurements collection of its distributed load injectors and probes. SelfBench [10] provides a virtualized and self-scalable load injection system to automatically detect the supportable upper bounder arrival rate of the system. SelfBench is based on CLIF. System MAR and corresponding resource utilization can be achieved by applying SelfBench.

Let us list terms and abbreviations used to describe our feasibility study method (see algorithms). To simplify explanation, we consider a three tier application.

Resource Allocation Scheme (RAS) is in the form of (FLAVOR1(NoI1), FLAVOR2(NoI2), FLAVOR3(NoI3)), where FLAVOR1(NoI1) denotes using NoI1 instances of FLAVOR1 on first tier.

CLIF(ARRIVAL_RATE, RAS) with result CAP/NO_CAP denotes do CLIF benchmark on RAS with ARRIVAL_RATE and get result that RAS can serve ARRIVAL_RATE(CAP) or not(NO_CAP).

SelfBench(RAS) with result MAX_ARRIVAL_RATE denotes do SelfBench on RAS and get the maximum capable arrival rate MAX_ARRIVAL_RATE of RAS.

Average resource utilization (e.g., Avg_RAM_UTIL(ARRIVAL_RATE) and Avg_CPU_UTIL(ARRIVAL_RATE)) can be achieved by both benchmarks.

BCF(TIER) denotes Best Choice Flavor for a given tier TIER . When adopting BCF(TIER), CPU utilization and RAM utilization are relatively balanced compared to non BCF(TIER).

MAXTH is the maximum arrival rate, which can be served by respecting the QoS target in SLA with a RAS of the biggest BCF(TIER) with minimum NoI.

ORAS(ARRIVAL_RATE) is an Optimized RAS, which is able to serve the arrival rate of ARRIVAL_RATE by respecting the QoS target with the minimum NoI and as small as possible size of BCF(TIER) according to SLA.

MFC(TIER,N,FLAVOR) = MAX_ARR_RATE denotes the maximum arrival rate can be served for a RAS with N instances of FLAVOR on TIER is *at most* MAX_ARR_RATE. So the integral capable arrival rate should be the minimum achievable MFC(TIER,N,FLAVOR) of all tiers. It is reasonable to assume that (1) is fulfilled.

$$MFC(TIER, N, FLAVOR) = N * MFC(TIER, 1, FLAVOR) \quad (1)$$

All possible MFC(TIER,1,FLAVOR) will be benchmarked or deduced for estimating the integral capable arrival rate of a RAS.

RAS(MFC(TIER,N,FLAVOR)) means a RAS which can serve maximum arrival rate of MFC(TIER,N,FLAVOR) with N instances of FLAVOR on TIER.

RAS(BCF(TIER),MIN_NoI) denotes a RAS with *the biggest* BCF and minimal NoI.

CAP_SLA_ReCst denotes the CAPable arrival rate according to SLA Resources Constraints.

COST(RAS) denotes the cost per charge unit for RAS and COST_AR_SLA is the minimal cost per charge unit for serving the arrival rate required in SLA.

IV. METHOD DESCRIPTION

Our method estimates the capable workload. Over estimation may lead to under provisioning, which means high probability of SLA violation. Under estimation leads to over provisioning, which means unnecessary cost. From the PaaS provider's perspective, low level under estimation is inevitable for signing a contract.

In this part, our SLA feasibility study method will be introduced according to the steps. For each step, we will formally describe our method and corresponding examples.

 ISBN: 978-1-61208-388-9

```
Require: FLAVOR, MIN_NoI
Ensure: BCF(TIER) for each tier.
function SELFBENCH_BCF( ORAS(FLAVOR), ORAS(MIN_NoI))
    SelfBench(S_gnrl(MIN_NoI),S_gnrl(MIN_NoI),S_gnrl(MIN_NoI))
    for each TIER do
        CPU_RAM_RATIO = Avg_CPU_UTIL/Avg_RAM_UTIL
        if CPU_RAM_RATIO ≥ 1.5 then
            BCF(TIER)=X_cpu
        else if CPU_RAM_RATIO ≤ 0.5 then
            BCF(TIER)=X_ram
        else
            BCF(TIER)=X_gnrl
        end if
    end for
    return BCF(TIER) for each tier
end function
```

Figure 1. Algorithm Find BCF(TIER)

```
Require: BCF(TIER), MIN_NoI
Ensure: MAXTH
function SELFBENCH_MAXTH(FLAVOR, MIN_NoI)
    SelfBench(RAS(BCF(TIER),MIN_NoI))
    return MAXTH = MAX_ARRIVAL_RATE
end function
```

Figure 2. Algorithm Find MAXTH

A. Step 1: Find BCF(TIER)

As described in the algorithm of Figure 1, we know BCF(TIER) if several kinds of flavors are allowed in SLA. Using non BCF(TIER) can be interesting when the maximum capability of BCF(TIER) is not sufficient, e.g., BCF(Apache) is X_gnrl, BCF(Jonas) is X_cpu and BCF(Mysql) is X_ram. So, ORAS(ARRIVAL_RATE) should have a RAS of (X_gnrl(1),X_cpu(2),X_ram(1)).

B. Step 2: Find MAXTH

As described in the algorithm of Figure 2, we do SelfBench on RAS (L_gnrl(1),L_cpu(2),L_ram(1)) with the biggest BCF(TIER) on each tier. MAXTH is 300 requests/s.

C. Step 3: ORAS(MAXTH) exploration

"new RAS" can be not existing because of SLA constraints. According to the algorithm of Figure 3, we check the resource utilization of doing SelfBench on (L_gnrl(1),L_cpu(2),L_ram(1)), and only Mysql tier is not yet saturated when arrival rate reaches MAXTH. So, the size of the Mysql tier flavor is decreased to get a "new RAS" (L_gnrl(1), L_cpu(2), M_ram(1)). CLIF benchmark result on the "new RAS" shows that MAXTH can be served, so we do CLIF benchmark on (L_gnrl(1),L_cpu(2),S_ram(1)), which indicate that MAXTH can not be served. So ORAS(MAXTH)=(L_gnrl,2L_cpu,M_ram).

D. Step 4: Exploring MFC of flavors used in ORAS(MAXTH)

According to the algorithm of Figure 4, ORAS(MAXTH) can be approximately seen as a RAS fitly allocated the resources to serve arrival rate of 300 requests/s. A RAS has one instance of L_gnrl on Apache tier, which is the same as ORAS(MAXTH)=300, can serve maximum arrival rate *at most* 300 requests/s. So MFC(Apache,1,L_gnrl)=300, MFC(Mysql,1,M_ram)=300 and MFC(Jonas,1,L_cpu)=MFC(Jonas,2,L_cpu)/2=MAXTH/2=150 according to equation 1.

```
Require: MIN_NoI, MAXTH, BCF(TIER), FLAVOR
Ensure: ORAS(MAXTH)
function FIND_ORAS_MAXTH( MIN_NoI, MAXTH, BCF(TIER), FLAVOR)
    ORAS(MAXTH) = NULL
    repeat
        TIER = 0
        repeat
            TIER ++
            Check Avg_RAM_UTIL, Avg_CPU_UTIL for TIER.
        until Both Avg_RAM_UTIL and Avg_CPU_UTIL are unsaturated
        if unsaturated TIER existing then
            "new RAS" is a RAS using smaller flavor on TIER.
            if "new RAS" existing then
                CLIF(MAXTH, "new RAS")
                if NO_CAP then
                    ORAS(MAXTH)="previous RAS"
                end if
            else
                ORAS(MAXTH)= "previous RAS"
            end if
        else
            ORAS(MAXTH)= "current RAS"
        end if
    until ORAS(MAXTH)!=NULL
    return ORAS(MAXTH)
end function
```

Figure 3. ORAS(MAXTH) exploration

```
Require: ORAS(MAXTH), MAXTH
Ensure: MFC(TIER,1,FLAVOR used in ORAS(MAXTH)
function SPECU_MFC_ORAS(ORAS(MAXTH), MAXTH)
    for each FLAVOR used by TIER in ORAS(MAXTH) do
        N= NoI for FLAVOR in TIER in ORAS(MAXTH)
        MFC(TIER,1,FLAVOR)=MAXTH/N
    end for
    return MFC(TIER,1,FLAVOR used in ORAS(MAXTH)
end function
```

Figure 4. Exploring MFC of flavors used in ORAS(MAXTH)

E. Step 5: Speculating MFC of flavors smaller than the one used in ORAS(MAXTH)

According to the algorithm of Figure 5, linear estimation as in Figure 6 can be used for deducing MFC of smaller flavor based on a bigger flavor. We consider both CPU and RAM. Comparing with the CPU and RAM of the flavor used in corresponding tier of ORAS(MAXTH), we get two ratios Ratio_CPU and Ratio_RAM, and we take the minimum ratio, which is lower or equals to 1. Then, MFC(tier,1,FLAVOR)= MIN_Ratio*MFC(tier,1,flavor in ORAS(MAXTH)). e.g., M_cpu allocated 5/3 of CPU and M_ram 1.67/6 of RAM. So, MIN_Ratio(M_cpu,

```
Require: FLAVOR, ORAS(MAXTH), MFC(TIER,1,flavor used in ORAS(MAXTH))
Ensure: MFC(TIER,1,smaller FLAVOR)
function SPECU_MFC_LESS( FLAVOR, ORAS(MAXTH), MFC(TIER,1,flavor used in ORAS(MAXTH)))
    for each FLAVOR in each TIER do
        Ratio_CPU(FLAVOR, flavor used in ORAS(MAXTH))
        Ratio_RAM(FLAVOR, flavor used in ORAS(MAXTH))
        MIN_Ratio = min(Ratio_CPU, Ratio_RAM)
        if MIN_Ratio ≤ 1 then
            MFC(TIER,1,FLAVOR_smlr_oras)= MIN_Ratio*MFC(TIER,1,flavor used in ORAS(MAXTH))
        end if
    end for
    return MFC(TIER,1,smaller FLAVOR)
end function
```

Figure 5. Speculating MFC of flavors smaller than the one used in ORAS(MAXTH)

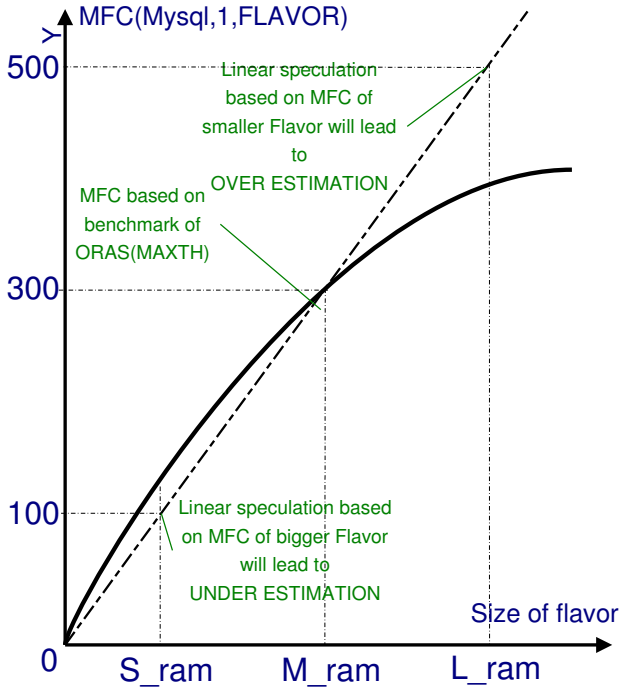


Figure 6. Under estimation and over estimation

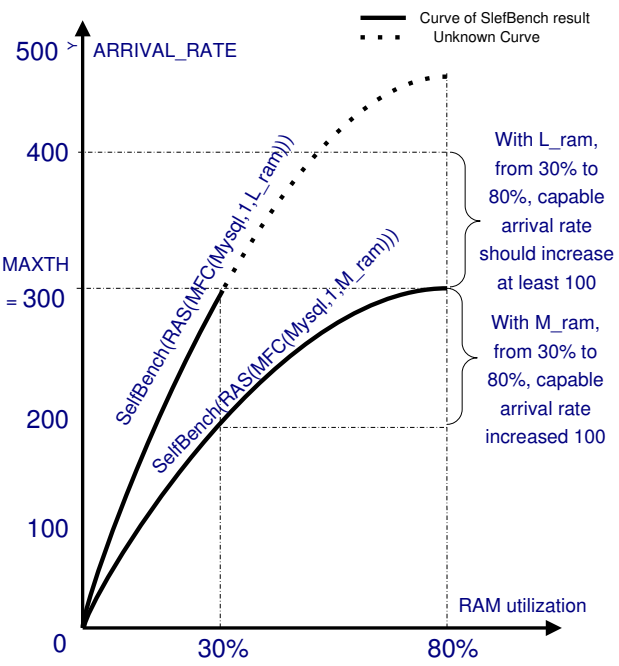


Figure 8. Linear estimation of bigger flavor MFC based on smaller flavor

Require: FLAVOR, ORAS(MAXTH), MIN_Ratio MFC(TIER,1,flavor used in ORAS(MAXTH))
Ensure: MFC(TIER,1,bigger FLAVOR).
1: **function** SPECU_MFC_GREATER(FLAVOR, ORAS(MAXTH), MIN_Ratio, MFC(TIER,1,flavor used in ORAS(MAXTH)))
2: SelfBench(ORAS(MAXTH))
3: **for** each FLAVOR MIN_Ratio $>$ 1 in each TIER **do**
4: Re_util= Avg_RAM_UTIL(MAXTH) for FLAVOR by checking benchmark report in Step2 or Step3
5: Arr_rate=ARRIVAL_RATE(Re_util) by checking SelfBench report in Step6.
6: augmentation = MAXTH - Arr_rate
7: MFC(TIER,1,bigger FLAVOR)=MAXTH + augmentation
8: **end for**
9: **return** MFC(TIER,1,bigger FLAVOR)
10: **end function**

Figure 7. Speculating MFC of flavors bigger than the one used in ORAS(MAXTH)

M_ram)=1.67/6. Since $1.67/6 < 1$, MFC(Mysql,1, M_cpu) = MIN_Ratio(M_cpu, M_ram) $*$MFC(Mysql,1,M_ram)$= 1.67/6 * 300 = 84$. Other MFC(TIER,1,FLAVOR) which have a MIN_Ratio ≤ 1 can be speculated similarly. If MIN_Ratio > 1, MFC(TIER,1,FLAVOR) can't be linearly deduced in the same way. Otherwise, overestimation will be introduced because of the concavity and convexity of the curve in Figure 6 [11].

F. Step 6: Speculating MFC of flavors bigger than the one used in ORAS(MAXTH)

If RAS1 and RAS2 use the same FLAVOR and NoI for TIER, the observable *overlapped* ARRIVAL_RATE of SelfBench(RAS1) and SelfBench(RAS2) results have approximately the same Avg_RAM_UTIL and Avg_CPU_UTIL SelfBench(L_gnrl(1),L_cpu(2),L_ram(1)) is done in Step2. Avg_RAM_UTILs and Avg_CPU_UTILs of Mysql tier for ARRIVAL_RATE from 0 to MAXTH are achievable and can

Require: MFC(TIER,1,FLAVOR), MAX_NoI
Ensure: AAR_RC_SLA.
1: **function** FIND_AAR_RC_SLA(MFC(TIER,1,FLAVOR, MAX_NoI)
2: **for** each TIER **do**
3: **for** each FLAVOR **do**
4: MFC(TIER,MAX_NoI,FLAVOR)= MFC(TIER,1,FLAVOR)* MAX_NoI
5: **end for**
6: MAX_MFC(TIER)= MAX{MFC(TIER,MAX_NoI,FLAVOR)}
7: **end for**
8: CAP_SLA_ReCst= MIN{MAX_MFC(TIER)}
9: **return** AAR_RC_SLA
10: **end function**

Figure 9. Resources constraints evaluation

be seen as the same for RAS(MFC(Mysql,1,L_ram)).

Then, we do SelfBench(ORAS(MAXTH)) and get the curve of Figure 8 for RAS(MFC(Mysql,1,M_ram)). The goal is to estimate ARRIVAL_RATE for RAS(MFC(Mysql,1,L_ram)) when Avg_RAM_UTIL is saturated (e.g., reaches 80%). This ARRIVAL_RATE is MFC(Mysql,1,L_ram). Avg_RAM_UTIL increases from 30% to 80%, ARRIVAL_RATE of RAS(MFC(Mysql,1,L_ram)) augmentation should be *more than* RAS(MFC(Mysql,1,M_ram))'s. Under estimation is made for MFC(Mysql,1,L_ram) 400 by taking the smaller augmentation of SelfBench(RAS(MFC(Mysql,1,M_ram))).

G. Step 7: Resources constraints evaluation

Until now, we have all the MFC(TIER,1,FLAVOR) I(d). The maximum capable arrival rate of all possible RAS, should be the minimum value, which is achieved by comparing the maximum MFC(TIER,MAX_NoI,FLAVOR) values in each tier, among all tiers. e.g., according to MAX_NoI Table I(b) from SLA and MFC(TIER,1,FLAVOR) Table I(d) achieved during benchmark, we know the MFC(tier, MAX_NoI, flavor) Table I(e). The maximum arrival rate can be served according

```
Require: MFC(TIER,1,FLAVOR), FLAVOR, MAR
Ensure: COST_AR_SLA.
function FIND_COST_AR_SLA(MFC(TIER,1,FLAVOR), FLAVOR, MAR)
    for each TIER do
        for each FLAVOR do
            MAX_NoI_MAR=⌈MAR/MFC(TIER, 1, FLAVOR)⌉
            COST(TIER,FLAVOR)= MAX_NoI_MAR * FLAVOR.COST
        end for
        MIN_COST(TIER)= min{COST(TIER,FLAVOR)}
    end for
    COST_AR_SLA= ∑MIN_COST(TIER)
    return COST_AR_SLA
end function
```

FIGURE 10. COST CONSTRAINTS EVALUATION

to resource constraints in SLA is MIN{600,600,900}=600, which is higher than SLA requirement(300 requests/s). So, the SLA can be achieved while fulfilling the resource constraints.

H. Step 8: Cost constraints evaluation

According to the algorithm of Figure 10, the cost for each tier with each flavor is shown in Table I(f). The minimum cost to serve MAR of 300 requests/s is 3+10.68+3 = 16.68(euro/s), which is higher than SLA cost constraint(maximum 10 euro/s). So SLA feasibility study refuses current version of SLA for cost constraints.

V. CONCLUSION

We proposed an SLA feasibility study method based on limited amount of benchmarking with application's Auto-scaling group architecture. The "what-if" evaluations answer the question that whether QoS target can be satisfied with the workload, resource and cost constraints stated in SLA. We experiment our method in the context of the OpenCloudware project [12] and on a small experimental set-up on which we are deploying a configurable virtualized application that we can stress in various ways to check the quality of derivations made from th benchmarks. Future work will be first to verify the accuracy of our proposal. Then, we will design our runtime controller to provide elasticity to virtualized applications using informations recorded during our benchmarking campaign while satsifying SLA.

ACKNOWLEDGMENT

The research described in this paper is supported by the OpenCloudware project [12].

REFERENCES

[1] W. Voorsluys, J. Broberg, and R. Buyya, Introduction to Cloud Computing. John Wiley & Sons, Inc., 2011, pp. 1–41. [Online]. Available: http://dx.doi.org/10.1002/9780470940105.ch1

[2] N. R. Herbst, S. Kounev, and R. Reussner, "Elasticity in Cloud Computing: What it is, and What it is Not," in Proceedings of the 10th International Conference on Autonomic Computing (ICAC 2013), San Jose, CA, June 24–28, 2013, pp. 23–27.

[3] D. Kyriazis, "Cloud Computing Service Level Agreements - Exploitation of Research Results," European Commission, Directorate General Communications Networs, Content and Technology Unit E2 - Software and Services, Cloud, Brussels, Belgium, Report/Study, Jun. 2013.

[4] P. Patel, A. Ranabahu, and A. Sheth, "Service level agreement in cloud computing," in Cloud Worshop, OOPSLA 2009, S. Arora and G. T. Leavens, Eds. Osrlando, FL, USA: ACM, Oct. 25-29 2009.

[5] G. Li, F. Pourraz, and P. Moreaux, "PSLA: a PaaS Level SLA Description Language," in Intercloud 2014, Boston, United States, Mar. 2014.

[6] R. W. Hockney, The Science of Computer Benchmarking, ser. Software, environments, tools. Society for Industrial and Applied Mathematics, 1996.

[7] A. Iosup, R. Prodan, and D. Epema, "Iaas cloud benchmarking: Approaches, challenges, and experience," in 5th Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS 2012, (SC),, Salt Lake City, UT, USA, Nov. 2012, invited paper.

[8] A. Alexandrov, E. Folkerts, K. Sachs, A. Iosup, V. Markl, and C. Tosun, "Benchmarking in the cloud: What it should, can, and cannot be," Aug 2012.

[9] B. Dillenseger, "Clif, a framework based on fractal for flexible, distributed load testing," annals of telecommunications, vol. 64, no. 1-2, 2009, pp. 101–120.

[10] A. Tchana et al., "A self-scalable and auto-regulated request injection benchmarking tool for automatic saturation detection," Cloud Computing, IEEE Transactions on, vol. 2, no. 3, July 2014, pp. 279–291.

[11] N. Gunther, Guerrilla Capacity Planning: A Tactical Approach to Planning for Highly Scalable Applications and Services. Springer, 2007.

[12] "OpenCloudware," [retrieved 02,2015], http://opencloudware.org/.

 ISBN: 978-1-61208-388-9

About Microservices, Containers and their Underestimated Impact on Network Performance

Nane Kratzke

Lübeck University of Applied Sciences, Center of Excellence CoSA
Lübeck, Germany
email: nane.kratzke@fh-luebeck.de

***Abstract*—Microservices are used to build complex applications composed of small, independent and highly decoupled processes. Recently, microservices are often mentioned in one breath with container technologies like Docker. That is why operating system virtualization experiences a renaissance in cloud computing. These approaches shall provide horizontally scalable, easily deployable systems and a high-performance alternative to hypervisors. Nevertheless, performance impacts of containers on top of hypervisors are hardly investigated. Furthermore, microservice frameworks often come along with software defined networks. This contribution presents benchmark results to quantify the impacts of container, software defined networking and encryption on network performance. Even containers, although postulated to be lightweight, show a noteworthy impact to network performance. These impacts can be minimized on several system layers. Some design recommendations for cloud deployed systems following the microservice architecture pattern are derived.**

Keywords–Microservice; Container; Docker; Software Defined Network; Performance

I. INTRODUCTION

Microservices are applied by companies like Amazon, Netflix, or SoundCloud [1] [2]. This architecture pattern is used to build big, complex and horizontally scalable applications composed of small, independent and highly decoupled processes communicating with each other using language-agnostic application programming interfaces (API). Microservice approaches and container-based operating system virtualization experience a renaissance in cloud computing. Especially container-based virtualization approaches are often mentioned to be a high-performance alternative to hypervisors [3]. *Docker* [4] is such a container solution, and it is based on operating system virtualization using Linux containers. Recent performance studies show only little performance impacts to processing, memory, network or I/O [5]. That is why *Docker* proclaims itself a ”lightweight virtualization platform” providing a standard runtime, image format, and build system for Linux containers deployable to any Infrastructure as a Service (IaaS) environment.

This study investigated the performance impact of Linux containers on top of hypervisor based virtual machines logically connected by an (encrypted) overlay network. This is a common use case in IaaS Cloud Computing being applied by popular microservice platforms like Mesos [6], CoreOS [7] or Kubernetes [8] (the reader may want to study a detailed analysis of such kind of platforms [9]). Nevertheless, corresponding performance impacts have been hardly investigated so far. Distributed cloud based microservice systems of typical complexity often use hypertext transfer protocol (HTTP) based and representational state transfer (REST) styled protocols to enable horizontally scalable system designs [10]. If these systems are deployed in public clouds, additional requirements for encrypted data transfer arise. There exist several open source projects providing such a microservice approach on top of IaaS provider specific infrastructures using this approach (e.g. Mesos, Kubernetes, CoreOS and more). These approaches are intended to be deployable to public or private IaaS infrastructures [9]. So in fact, these approaches apply operating system virtualization (containers) on top of hypervisors (IaaS infrastructures). Although almost all of these microservice frameworks rely heavily on the combination of containerization on top of hypervisors, and some of these approaches introduce additional overlay networking and data encryption layers, corresponding performance impacts have been hardly analyzed so far. Most performance studies compare container performance with virtual machine performance but not container performance on top of virtual machines (see Felter at al. [5] for a typical performance study).

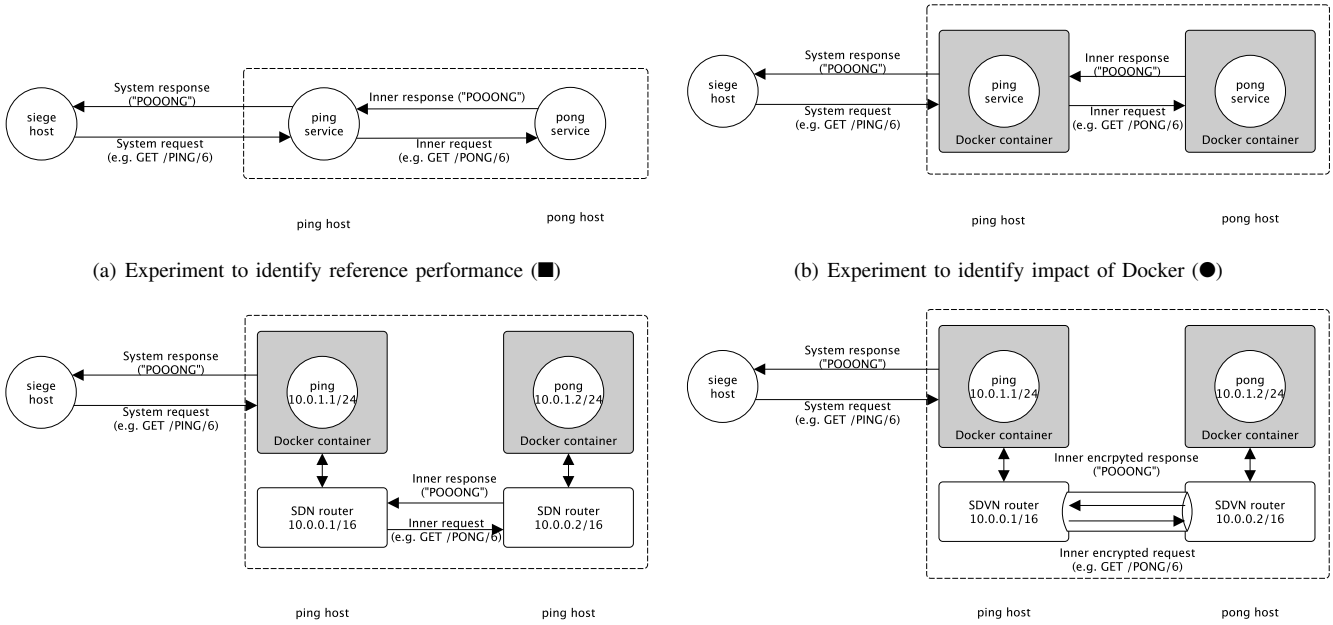
Because overlay networks are often reduced to distributed hashtable (DHT) or peer-to-peer approaches, this paper uses the term software defined virtual networks (SDVN). SDVNs, in the understanding of this paper, are used to provide a logical internet protocol (IP) network for containers on top of IaaS infrastructures.

Section II presents related work about state-of-the-art container approaches and SDVN solutions. Section III explains the experiment design to identify performance impacts of containers, SDVNs and encryption. The benchmark tooling [11] and the performance data collected is provided online [12]. Resulting performance impacts are discussed in Section IV. Derived design recommendations to minimize performance impacts on application, overlay network and IaaS infrastructure layer are presented in concluding Section V.

II. RELATED WORK

Although container based operating system virtualization is postulated to be a scalable and high-performance alternative to hypervisors, hypervisors are the standard approach for IaaS cloud computing [3]. Felter et al. provided a very detailed analysis on CPU, memory, storage and networking resources to explore the performance of traditional virtual machine deployments, and contrast them with the use of Linux containers provided via *Docker* [5]. Their results indicate that benchmarks that have been run in a *Docker* container,

 ISBN: 978-1-61208-388-9



(a) Experiment to identify reference performance (■)

(b) Experiment to identify impact of Docker (●)

(c) Experiment to identify impact of SDVN (▲)

(d) Experiment to identify impact of encryption (▼)

Figure 1. Experiments

show almost the same performance (floating point processing, memory transfers, network bandwidth and latencies, block I/O and database performances) like benchmarks run on "bare metal" systems. Nevertheless, Felter et al. did not analyze the impact of containers on top of hypervisors.

Although there exist several SDVN solutions for *Docker*, only one open source based SDVN has been identified, which is able to encrypt underlying data transfers: *Weave* [13]. That is why other SDVN approaches for *Docker* like *flannel* [14] or *docknet* [15] are not covered by this study. Pure virtual local area network (VLAN) solutions like *Open vSwitch (OVS)* [16] are not considered, because *OVS* is not to be designed for operating system virtualization. So, *weave* remained as the only appropriate SDVN candidate for this study. But the author is confident that this will change in the future and more encryptable SDVN solutions will arise.

Weave creates a network bridge on *Docker* hosts to enable SDVN for *Docker* containers. Each container on a host is connected to that bridge. A *weave* router captures Ethernet packets from its bridge-connected interface in promiscuous mode. Captured packets are forwarded over the user datagram protocol (UDP) to *weave* router peers running on other hosts. These UDP "connections" are duplex, can traverse firewalls and can be encrypted.

To analyze the performance impact of containers, software defined networks and encryption, this paper considered several contributions on cloud related network performance analysis (see [17], [18], [19], [20], [21], [22]). But none of these contributions focused explicitly on horizontally scalable systems with HTTP-based and REST-like protocols. To address this common use case for microservice architectures, this paper proposes the following experiment design.

III. EXPERIMENT DESIGN

This study analyzed the network performance impact of container, SDVN and encryption layers on the performance impact of distributed cloud based systems using HTTP-based REST-based protocols. Therefore, five experiments have been designed (see Figure 1). The analyzed *ping-pong* system relied on a REST-like and HTTP-based protocol to exchange data. Apachebench [23] was used to collect performance data of the *ping-pong* system. *Siege*, *ping* and *pong* servers have been deployed to the Amazon Web Services (AWS) IaaS infrastructure on a m3.medium instance type. Experiments have been run in eu-west-1c availability zone (Ireland). The *siege* server run the apachebench benchmark. The *ping* and *pong* application were developed using Googles Dart programming language [24]. To understand the performance impact of containers, SDVN and encryption to network performance, the study analyzed the data transfer rate $trans(m)$ of m byte long messages.

■ The **reference experiment** shown in Figure 1(a), was used to collect reference performance data of the *ping-pong* system deployed to different virtual machines interacting with a REST-like and HTTP based protocol. No containers, SDVN or encryption were used in this experiment. Further experiments added a container, a SDVN and an encryption layer to measure their impact on network performance. A *ping* host interacts with a *pong* host to provide its service. Whenever the *ping* host is requested by *siege* host, the *ping* host relays the original request to the *pong* host. The *pong* host answers the request with a response message. The *siege* host can define the inner message and system response message length by query. All requests are performed using the HTTP protocol. The *siege* host is used to run several *apachebench* benchmark runs with increasing requested message sizes to measure the system performance of the *ping-pong* system. This paper refers

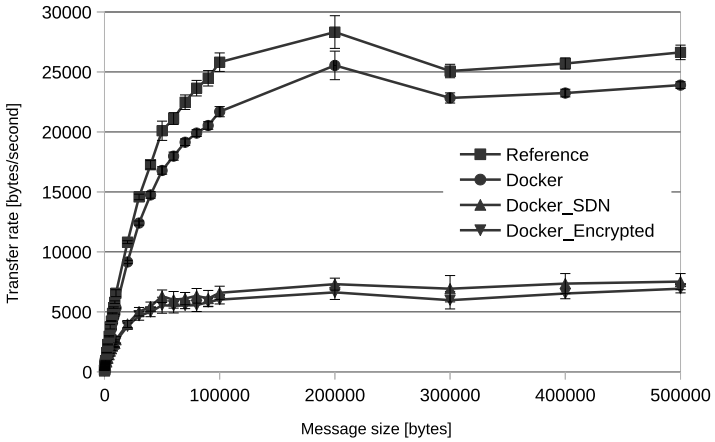


Figure 2. Absolute performance impact on transfer rates

to measured transfer rates for a message size of m (bytes) of this experiment as $trans_{\square}(m)$. These absolute values are presented in Figure 2.

● The intent of the **Docker experiment** was to figure out the impact of an additional container layer to network performance (Figure 1(b)). So, the *ping* and *pong* services are provided as containers to add an additional container layer to the reference experiment. Every performance impact must be due to this container layer. The measured transfer rates are denominated as $trans_{\bigcirc}(m)$. The impact of containers on transfer rates is calculated as follows and presented in Figure 3(a):

$$\bigcirc_{trans}(m) = \frac{trans_{\bigcirc}(m)}{trans_{\square}(m)} \tag{1}$$

▲ The intent of the **SDVN experiment** shown in Figure 1(c) was to figure out the impact of an additional SDVN layer to network performance. This experiment connects *ping* and *pong* containers by a SDVN. So, every data transfer must pass the SDVN solution between *ping* and *pong*. This paper refers to measured transfer rates for a message size of m (bytes) of this experiment as $trans_{\Delta}(m)$. The impact of SDVN on transfer rates for a message size m is calculated as follows and presented in Figure 3(a):

$$\Delta trans(m) = \frac{trans_{\Delta}(m)}{trans_{\square}(m)} \tag{2}$$

▼ The **encryption experiment** (see Figure 1(d)) figured out the impact of an additional data encryption layer on top of a SDVN layer. Additionally, this experiment encrypts the SDVN network. Corresponding measured transfer rates are denominated as $trans_{\nabla}(m)$. The impact of SDVN on transfer rates for a message size m is calculated as follows and is presented in Figure 3(a):

$$\nabla trans(m) = \frac{trans_{\nabla}(m)}{trans_{\square}(m)} \tag{3}$$

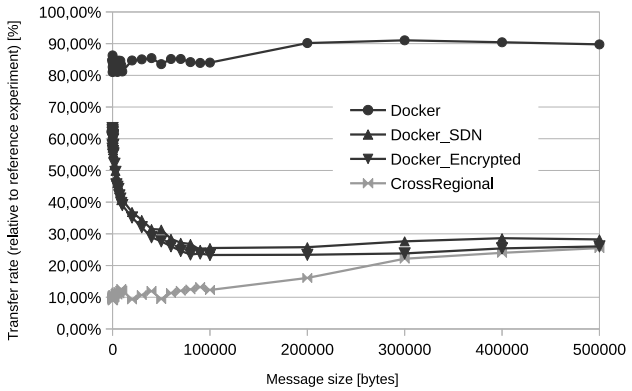
⋈ The intent of **cross-regional experiment** was to figure out the impact of an cross-regional deployment to network performance. Although this was not the main focus of the study, this use case has been taken into consideration to generate a more graspable performance impact understanding for the reader. The setting has been the same as in Figure 1(a), except that the *ping* and *pong* hosts were deployed to different regions of the AWS infrastructure. *ping* (and the *siege* host) were deployed to the AWS region eu-west-1c (EU, Ireland) and the *pong* host was deployed to AWS region ap-northeast-1c (Japan, Tokyo). Data from this experiment is only used to compare container, SDVN and encrypted SDVN performance with a cross-regional performance impact in a qualitative manner. A cross-regional impact might be more intuitively graspable for the reader. The cross-regional impact on transfer rates is presented in Figure 3(a) as a lightgrey line.

IV. DISCUSSION OF RESULTS

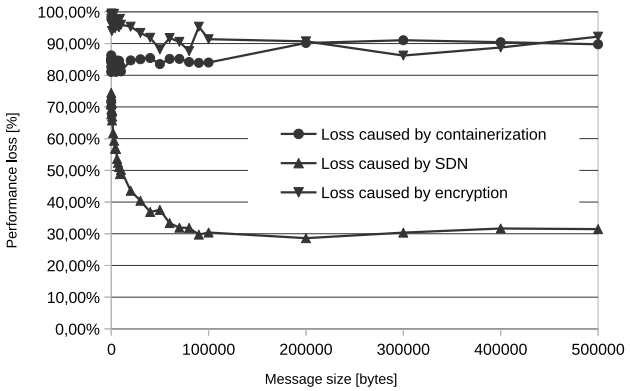
The results presented are based on more than 12 hours of benchmark runs, which resulted in a transfer of over 316GB of data requested by more than 6 million HTTP requests (see Table I). Reference and Docker experiments show only minor standard deviations. The maximum deviations were measured for very small message sizes (10 and 20 byte). Standard deviations increased with the introduction of SDVN. So, the collected deviation data indicates that the experiment setting produces reliable data (increasing deviations of SDVN experiment have to do with the technical impacts of SDVNs, they are not due to experiment design and will be discussed by this paper).

● *Container impact:* Containers are stated to be lightweight and to have only negligible performance impacts [5]. The *Docker* experiment shows a somewhat different picture. A non negligible performance loss can be identified for data transfer rates (see Figure 2). An additional container layer on top of a bare virtual machine reduces the performance to 80% (for message sizes smaller than 100 kBytes) to 90% (for message sizes greater than 100kBytes). The study also included a cross-zone deployment (similar to the cross-region deployment but in two different availability zones of the same AWS region). It turned out that a cross-zone deployment

 ISBN: 978-1-61208-388-9



(a) comparison of transfer rates (100% means no loss)



(b) Resulting transfer losses (100% means no loss)

Figure 3. Relative comparison of performance indicators

shows almost the same performance like an one-zone deployment (reference experiment). Containers show a significant higher performance impact than cross-zone deployments in IaaS clouds. So, compared with cross-zone deployments (non-measurable effects), we have to say that containers have measurable (non-negligible) impacts to network performance.

▲ *SDVN impact:* The impact of analyzed SDVN solution *weave* reduces data transfer rates from 25 kB/s to about 7,5kB/s (see Figure 2). Figure 3a shows the relative performance of the SDVN experiment. SDVN experiments show only 60% performance of the reference experiment for small message sizes going down to about 25% performance for message sizes greater than 100kB. The SDVN experiment shows a comparable performance impact like a cross-regional deployment (for big message sizes, Ireland ↔ Japan). *Weave* SDVN routers are provided as *Docker* containers. The SDVN experiment measures the performance impact of containerization **and** SDVN. To identify the pure SDVN effect, the reader has to compare SDVN data (Δ) relative to the performance data of containers ($\bigcirc$) to exclude the container effects (see Figure 3b).

$$impact_{\Delta}(m) = \frac{trans_{\Delta}(m)}{trans_{\bigcirc}(m)} \tag{4}$$

To avoid container losses, SDVN solutions should be provided directly on the host and not in a containerized form. This should reduce the performance loss about 10% to 20%. Furthermore, it is noted that tests were running on a single-core virtual machine (m3.medium type). In saturated network load situations, the *weave* router contends for CPU with the application processes, so it will saturate faster compared with Reference or *Docker* experiment where the network is handled by the hypervisor and physical network, outside of such contention. This effect explains the severe performance impacts shown in Figure 3. That lead us to the design conclusion that SDVN solutions should always run on multi-core systems to avoid severe performance impacts due to contention.

▼ *Encryption impact:* Additional encryption shows only minor impacts to transfer rates compared with SDVN without encryption (see Figure 2). So, most of the performance losses are due to SDVN and not because of encryption. To identify the pure encryption effect, encrypted SDVN data (∇) has to be compared relative to the performance data of SDVN experiment (Δ).

TABLE I. RELATIVE STANDARD DEVIATIONS OF MEASURED TRANSFER RATES

		%RSD		
Experiment	**Data**	**Min**	**Avg**	**Max**
Reference	90 GB	0,9	2,9	15,9
Cross Regional	19 GB	0,9	14,9	28,7
Docker	57 GB	0,8	2,0	10,3
Docker_SDVN	75 GB	0,4	11,3	21,2
Docker_Encrypted	75 GB	0,5	10,9	16,2

$$impact_{\nabla}(m) = \frac{trans_{\nabla}(m)}{trans_{\Delta}(m)} \tag{5}$$

Encryption reduces the transfer performance down to about 90% compared with the transfer rates of non encrypted data transfers. For smaller message sizes this negative effect of encryption gets even more and more negligible. In other words, especially for small message sizes encryption is not a substantial performance killer compared to SDVN impact (see Figure 3b). For bigger message sizes the data transfer performance impact of encryption is comparable to containerization.

V. CONCLUSION

This study analyzed performance impact of containers, overlay networks and encryption to overall network performance of HTTP-based and REST-like services deployed to IaaS cloud infrastructures. Obviously, our conclusions should be cross checked with other SDVN solutions for containers. The provided data [12] and benchmarking tools to apply the presented methodology [11] can be used as benchmark for that purpose. Nevertheless, some of the study results can be used to derive some design recommendations for cloud deployed HTTP-based and REST-like systems of general applicability.

Although **containers** are stated to be lightweight [3] [5], this study shows that container impact on network performance

 ISBN: 978-1-61208-388-9

is not negligible. Containers show a performance impact of about 10% to 20%. The impact of **overlay networks** can be even worse. The analyzed SDVN solution showed a performance impact of about 30% to 70%, which is comparable to a cross regional deployment of a service between Ireland and Japan. **Encryption** performance loss is minor, especially for small message sizes.

The results show that performance impacts of overlay networks can be minimized on several layers. On **application layer** message sizes between system components should be minimized whenever possible. Network performance impact gets worse with increasing message sizes. On **overlay network layer** performance could be optimized by 10% to 20% by providing SDVN router applications directly on the host (in a not containerized form, because 10% to 20% are due to general container losses). On **infrastructure layer**, the SDVN routers should be deployed to multi core virtual machines to avoid situations, where SDVN routers contend for CPU with application processes.

So containers, which are often mentioned to be lightweight, are not lightweight under all circumstances. Nevertheless, the reader should not conclude to avoid container and SDVN technologies in general. Container and SDVN technologies provide more flexibility and manageability in designing complex horizontally scalable distributed cloud systems. And there is nothing wrong about flexibility and manageability of complex systems. That is why container solutions like Docker and microservice approaches regain so much attention recently. But container and SDVN technologies should be always used with above mentioned performance implications in mind.

ACKNOWLEDGMENT

This study was funded by German Federal Ministry of Education and Research (Project Cloud TRANSIT, 03FH021PX4). The author thanks Lübeck University (Institute of Telematics) and fat IT solution GmbH (Kiel) for their support of Cloud TRANSIT. The author also thanks Bryan Boreham of zett.io for checking our data of zett.io's *weave* solution (which might show now better results than the analyzed first version of *weave*).

REFERENCES

[1] M. Fowler and J. Lewis. Microservices. Last access 17th Feb. 2015. [Online]. Available: http://martinfowler.com/articles/microservices.html [retrieved: March, 2014]

[2] S. Newman, Building Microservices. O'Reilly and Associates, 2015.

[3] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors," SIGOPS Oper. Syst. Rev., vol. 41, no. 3, Mar. 2007, pp. 275–287.

[4] Docker. Last access 17th Feb. 2015. [Online]. Available: https://docker.com

[5] W. Felter, A. Ferreira, R. Rajamony, and J. Rubio, "An updated performance comparison of virtual machines and linux containers," IBM Research Division, Austin Research Laboratory, Tech. Rep., 2014.

[6] Mesos. Last access 17th Feb. 2015. [Online]. Available: https://mesos.apache.org

[7] Coreos. Last access 17th Feb. 2015. [Online]. Available: https://coreos.com

[8] Kubernetes. Last access 17th Feb. 2015. [Online]. Available: https://github.com/GoogleCloudPlatform/kubernetes

[9] N. Kratzke, "A lightweight virtualization cluster reference architecture derived from open source paas platforms," Open Journal of Mobile Computing and Cloud Computing (MCCC), vol. 1, no. 2, 2014, pp. 17–30.

[10] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[11] N. Kratzke. Ping pong - a distributed http-based and rest-like ping-pong system for test and benchmarking purposes. Last access 17th Feb. 2015. [Online]. Available: https://github.com/nkratzke/pingpong

[12] ——. Collected performance data. Last access 17th Feb. 2015. [Online]. Available: https://github.com/nkratzke/sdvn-impact-database

[13] Weave. Last access 17th Feb. 2015. [Online]. Available: https://github.com/zettio/weave

[14] Flannel. Last access 17th Feb. 2015. [Online]. Available: https://github.com/coreos/flannel

[15] Docknet. Last access 17th Feb. 2015. [Online]. Available: https://github.com/helander/docknet

[16] Open vswitch. Last access 17th Feb. 2015. [Online]. Available: http://openvswitch.org

[17] D. Mosberger and T. Jin, "Httperf—a tool for measuring web server performance," SIGMETRICS Perform. Eval. Rev., vol. 26, no. 3, Dec. 1998, pp. 31–37. [Online]. Available: http://doi.acm.org/10.1145/306225.306235

[18] G. Memik, W. H. Mangione-Smith, and W. Hu, "Netbench: A benchmarking suite for network processors," in Proceedings of the 2001 IEEE/ACM International Conference on Computer-aided Design, ser. ICCAD '01. Piscataway, NJ, USA: IEEE Press, 2001, pp. 39–42.

[19] J. Verdú, J. Garcí, M. Nemirovsky, and M. Valero, "Architectural impact of stateful networking applications," in Proceedings of the 2005 ACM Symposium on Architecture for Networking and Communications Systems, ser. ANCS '05. New York, NY, USA: ACM, 2005, pp. 11–18.

[20] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. Wright, "Performance analysis of high performance computing applications on the amazon web services cloud," in Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, Nov 2010, pp. 159–168.

[21] G. Wang and T. Ng, "The impact of virtualization on network performance of amazon ec2 data center," in INFOCOM, 2010 Proceedings IEEE, March 2010, pp. 1–9.

[22] D. Jayasinghe, S. Malkowski, J. LI, Q. Wang, Z. Wang, and C. Pu, "Variations in performance and scalability: An experimental study in iaas clouds using multi-tier workloads," Services Computing, IEEE Transactions on, vol. 7, no. 2, April 2014, pp. 293–306.

[23] Apachebench. Last access 17th Feb. 2015. [Online]. Available: http://httpd.apache.org/docs/2.2/programs/ab.html

[24] Dart. Last access 17th Feb. 2015. [Online]. Available: https://www.dartlang.org/

 ISBN: 978-1-61208-388-9